FRANK-WOLFE METHODS FOR OPTIMIZATION AND MACHINE LEARNING

A Dissertation
Presented to
The Academic Faculty

By

Cyrille W. Combettes

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Industrial and Systems Engineering

Georgia Institute of Technology

May 2021

FRANK-WOLFE METHODS FOR OPTIMIZATION AND MACHINE LEARNING

Thesis committee:

Dr. Alexandre d'Aspremont
Départment d'Informatique
CNRS and École Normale Supérieure

Dr. Arkadi S. Nemirovski
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Swati Gupta
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Sebastian Pokutta (advisor)
School of Industrial and Systems Engineering and Institute of Mathematics and Department for AI in Society, Science, and Technology
Georgia Institute of Technology and Technische Universität Berlin and Zuse Institute Berlin

Dr. Guanghui Lan
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date approved: April 16, 2021

ACKNOWLEDGMENTS

I would like to express my sincere appreciation to the many people that have made this Ph.D. journey possible.

To Sebastian Pokutta: It has been a privilege to be your student during these three years. You have been a trustworthy and mindful mentor. Your all-around support has been crucial and I am very happy to have conducted my Ph.D. under your guidance.

To Alexandre d'Aspremont, Swati Gupta, Guanghui Lan, and Arkadi Nemirovski: Thank you for serving on my Ph.D. committee. It has been a real pleasure to defend my thesis before you and I have enjoyed our discussion very much.

To Christoph Spiegel: It was fun to collaborate with you on a paper. I am also indebted to you for helping me with registration in Berlin.

To Alejandro Carderera: My go-to guy at Georgia Tech. We have taken all our classes together, traveled to Tokyo, Toronto, and Berlin for workshops and conferences, helped each other solve administrative headaches, and much more.

To Makram Chahine and Robert Gan: My experience as a graduate student at Georgia Tech would not be full of vibrant memories without you. Thanks also to the CRC, for all the time we have spent in its awesome facilities.

To Pierre Léger, Paul Gozlan, and especially Nicolas Tosel: Not a day goes by when I do not realize the quality of the education each of you has provided me with at École Élémentaire d'Alésia, Collège Jean Moulin, and Lycée Louis-le-Grand. I am very grateful for your inspiring teaching and dedicated mentoring.

To my parents: For everything.

My work was supported by the National Science Foundation under CAREER Award CMMI-1452463, the Deutsche Forschungsgemeinschaft through the Cluster of Excellence

MATH+, and the Research Campus MODAL funded by the German Federal Ministry of Education and Research under grant numbers 05M14ZAM and 05M20ZBM.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

In Chapter 2, we present the Frank-Wolfe algorithm (FW) and all necessary background material. We explain the projection-free and sparsity properties of the algorithm, provide motivation for real-world problems, and analyze the convergence rates and a lower bound on the complexity.

In Chapter 3, we review the complexity bounds of linear minimizations and projections on several sets commonly used in optimization, providing a rigorous support to the use of FW. We also propose two methods for projecting onto the ℓ_p-ball and the Birkhoff polytope respectively, and we analyze their complexity. Computational experiments for the ℓ_1-ball and the nuclear norm-ball are presented.

In Chapter 4, we identify the well-known drawback in FW, a naive zig-zagging phenomenon that slows down the algorithm. In response to this issue, we propose a boosting procedure generating descent directions better aligned with the negative gradients and preserving the projection-free property. Although the method is relatively simple and intuitive, it provides significant computational speedups over the state of the art on a variety of experiments.

In Chapter 5, we address the large-scale finite-sum optimization setting arising in many tasks of machine learning. Based on a sliding technique, we propose a generic template to integrate adaptive gradients into stochastic Frank-Wolfe algorithms in a practical way. Computational experiments on standard convex optimization problems and on the nonconvex training of neural networks demonstrate that the blend of the two methods is successful.

Both developments in Chapters 4 and 5 are motivated by the projection-free property of FW. In Chapter 6, we leverage the natural sparsity of the iterates generated by FW and study an application to the approximate Carathéodory problem. We show that FW generates a simple solution to the problem and that with no modification of the algorithm, better cardinality bounds can be established using existing convergence analysis of FW in

different scenarios. We also consider a nonsmooth variant of FW.

In Chapter 7, we carry on with the sparsity property and we consider an extension of the Frank-Wolfe algorithm to the unconstrained setting. It addresses smooth convex optimization problems over the linear span of a given set and resembles the matching pursuit algorithm. We propose a blending method that combines fast convergence and high sparsity of the iterates. Computational experiments validate the purpose of our method.

CHAPTER 1
INTRODUCTION

1.1 Optimization with constraints

Optimization is a core component of machine learning for formulating, analyzing, and solving many problems of practical interest. It drives the theoretical methodology and the engineering setup to address problems in the real-world and establishes the interplay between them. Since the majority of state-of-the-art machine learning models require very vast amounts of data to be trained, it is essential that modern optimization methods blend a mix of sharp worst-case guarantees and strong average-case, i.e., practical, performance. These are driven by analytical-oriented and practice-aware algorithmic designs respectively.

In this thesis, we consider optimization problems with constraints. Constraints play an essential role in translating the key problem information into its mathematical formulation or in enforcing desired properties into the optimization solution. It is the way to input human knowledge into the machine. The latter has been popularized in signal processing, where the ℓ_2-norm addresses overfitting, the ℓ_1-norm and the nuclear norm recover sparse and low-rank solutions respectively, and, e.g., the total variation distance is used for denoising. New areas of machine learning also heavily rely on a constraint set, e.g., to enforce fairness in classification models or to train neural networks that are robust to adversarial perturbations.

We are interested in the constrained optimization problem

$$\min_{x \in \mathcal{C}} f(x),$$

where $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set and $f \colon \mathbb{R}^n \to \mathbb{R}$ is a smooth function. This thesis is focused on the Frank-Wolfe algorithm (FW) [43], a.k.a. conditional gradient algorithm

[85], a simple first-order method for constrained optimization. FW has three principal advantages which have made the method very successful in different applications:

(i) it is simple to implement,

(ii) it does not require projections back onto the constraint set to ensure feasibility,

(iii) it generates iterates that are sparse with respect to the vertices of the constraint set.

By item (i), we mean that FW is either very easy to tune or does not require any tuning at all to be executed. This shows that its practical performance is very closely related to its theoretical analysis. Item (ii) is a seminal property which makes FW belong to a class of first-order methods of its own. The amount and complexity of real-world information we are willing to include into the constraint set is balanced by the cost of handling it in the optimization phase, so it is highly valuable that FW is able to handle the constraint set at a low (per-iteration) cost. Item (iii) finds interesting applications in generating sparse solutions for specific optimization tasks.

1.2 Notation and definitions

We consider the standard Euclidean space $(\mathbb{R}^n, \langle \cdot, \cdot \rangle)$ equipped with an arbitrary norm $\| \cdot \|$. Note that the ambient space $\mathbb{R}^n$ could be replaced by the space of matrices $\mathbb{R}^{m \times n}$.

1.2.1 Basics

Definition 1.1. *For every $i, j \in \mathbb{N}$ such that $i \leqslant j$, the brackets $[\![i, j]\!]$ denote the set of integers between (and including) i and j.*

Definition 1.2. *For all $x \in \mathbb{R}^n$ and $i, j \in [\![1, n]\!]$ such that $i \leqslant j$, $[x]_i$ denotes the ith entry of x and $[x]_{i:j} = ([x]_i, \ldots, [x]_j)^\top \in \mathbb{R}^{j-i+1}$.*

Definition 1.3. *Let $p \in [1, +\infty[$. The ℓ_p-norm is*

$$\|\cdot\|_p \colon x \in \mathbb{R}^n \mapsto \left(\sum_{i=1}^{n} |[x]_i|^p \right)^{1/p}.$$

The ℓ_∞-norm is

$$\|\cdot\|_\infty \colon x \in \mathbb{R}^n \mapsto \max_{i \in [\![1,n]\!]} |[x]|_i.$$

Definition 1.4. *Let $x \in \mathbb{R}^n$ and $r > 0$. The (closed) ball of radius r centered at x is*

$$\mathcal{B}(x, r) = \{y \in \mathbb{R}^n \mid \|y - x\| \leqslant r\}.$$

Definition 1.5. *The dual norm of $\|\cdot\|$ is*

$$\|\cdot\|_* \colon y \in \mathbb{R}^n \mapsto \sup_{\|x\| \leqslant 1} \langle x, y \rangle.$$

Lemma 1.6 (Cauchy-Schwarz)**.** *For all $x, y \in \mathbb{R}^n$,*

$$\langle x, y \rangle \leqslant \|x\| \|y\|_*.$$

Definition 1.7. *Let $\mathcal{A}$ and $\mathcal{B}$ be two sets of $\mathbb{R}^n$. Then $\mathcal{A}$ is included in $\mathcal{B}$, and we write $\mathcal{A} \subset \mathcal{B}$, if for all $x \in \mathcal{A}$, it holds $x \in \mathcal{B}$. In particular, if $\mathcal{A} = \mathcal{B}$ then $\mathcal{A} \subset \mathcal{B}$.*

1.2.2 Sets

Definition 1.8. *A polytope is a nonempty and bounded polyhedral set.*

Definition 1.9. *A set $\mathcal{C} \subset \mathbb{R}^n$ is convex if for all $x, y \in \mathcal{C}$ and $\gamma \in [0, 1]$,*

$$(1 - \gamma)x + \gamma y \in \mathcal{C}.$$

Thus, the line segment between any two points of a convex set is contained in the set. This assumption can be extended to that of uniform convexity, which states that for any point in the line segment, there exists a ball centered at the point and contained in the set. Note that the radius of the ball at the extremities of the line segment is not required to be nonzero. For example, a polytope is convex but is not uniformly convex.

Definition 1.10. *A set $\mathcal{C} \subset \mathbb{R}^n$ is (α, q)-uniformly convex if $\alpha, q > 0$ and for all $x, y \in \mathcal{C}$, $\gamma \in [0, 1]$, and $z \in \mathbb{R}^n$ with $\|z\| = 1$,*

$$(1-\gamma)x + \gamma y + (1-\gamma)\gamma\alpha\|x-y\|^q z \in \mathcal{C}.$$

Definition 1.11. *A set $\mathcal{C} \subset \mathbb{R}^n$ is α-strongly convex if it is $(\alpha, 2)$-uniformly convex.*

1.2.3 Functions

Definition 1.12. *A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is G-Lipschitz continuous if $G > 0$ and for all $x, y \in \mathbb{R}^n$,*

$$|f(y) - f(x)| \leqslant G\|y - x\|.$$

Definition 1.13. *A differentiable function $f\colon \mathbb{R}^n \to \mathbb{R}$ has L-Lipschitz continuous gradient if $L > 0$ and for all $x, y \in \mathbb{R}^n$,*

$$\|\nabla f(y) - \nabla f(x)\|_* \leqslant L\|y - x\|.$$

Definition 1.14. *A differentiable function $f\colon \mathbb{R}^n \to \mathbb{R}$ is L-smooth if $L > 0$ and for all $x, y \in \mathbb{R}^n$,*

$$f(y) \leqslant f(x) + \langle y - x, \nabla f(x)\rangle + \frac{L}{2}\|y - x\|^2.$$

Lemma 1.15 (Descent)**.** *Let $f\colon \mathbb{R}^n \to \mathbb{R}$. If f has L-Lipschitz continuous gradient, then f is L-smooth. The converse is true if f is convex.*

Definition 1.16. *A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is convex if for all $x, y \in \mathbb{R}^n$ and $\gamma \in [0,1]$,*

$$f((1-\gamma)x + \gamma y) \leqslant (1-\gamma)f(x) + \gamma f(y).$$

Lemma 1.17 (Jensen)**.** *Let $f\colon \mathbb{R} \to \mathbb{R}$ be a convex function and X be a random variable on an interval $I \subset \mathbb{R}^n$ such that $\mathsf{E}[|X|], \mathsf{E}[|f(X)|] < +\infty$. Then*

$$f(\mathsf{E}[X]) \leqslant \mathsf{E}[f(X)].$$

Definition 1.18. *A differentiable function $f\colon \mathbb{R}^n \to \mathbb{R}$ is S-strongly convex if $S > 0$ and for all $x, y \in \mathbb{R}^n$,*

$$f(y) \geqslant f(x) + \langle y - x, \nabla f(x)\rangle + \frac{S}{2}\|y - x\|^2.$$

Definition 1.19. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a convex set. A differentiable function $f\colon \mathbb{R}^n \to \mathbb{R}$ is μ-gradient dominated on $\mathcal{C}$ if $\mu > 0$ and for all $x \in \mathcal{C}$,*

$$f(x) - \min_{\mathcal{C}} f \leqslant \frac{\|\nabla f(x)\|_*^2}{2\mu}.$$

Note that if f is gradient dominated on $\mathbb{R}^n$, then it is gradient dominated on any convex set $\mathcal{C} \subset \mathbb{R}^n$. The gradient dominated property is also commonly referred to as the Polyak-Łojasiewicz inequality [114, 91]. It is a *local* condition, weaker than that of strong convexity (Fact 1.20), but it can still provide linear convergence rates for non-strongly convex functions. For example, the least squares loss $x \in \mathbb{R}^n \mapsto \|Ax - b\|_2^2$ where $A \in \mathbb{R}^{m\times n}$ and $\operatorname{rank}(A) = m < n$ is not strongly convex, however it is gradient dominated [45]. See also the Kurdyka-Łojasiewicz inequality [76, 91] for a generalization to nonsmooth

optimization [12].

Fact 1.20. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be S-strongly convex. Then f is $S/4$-gradient dominated. If $\|\cdot\|$ is the ℓ_2-norm, then f is S-gradient dominated.*

Proof. The function f is strongly convex hence it has a unique minimizer, which we denote by $x^* \in \mathbb{R}^n$. Let $x \in \mathbb{R}^n \backslash \{x^*\}$. We start with the case of a general norm $\|\cdot\|$. By optimality of x^*, we have

$$\langle x - x^*, \nabla f(x^*) \rangle \geqslant 0,$$

so, by strong convexity,

$$f(x) - f(x^*) \geqslant \langle x - x^*, \nabla f(x^*) \rangle + \frac{S}{2}\|x - x^*\|^2 \geqslant \frac{S}{2}\|x - x^*\|^2.$$

Thus, by convexity and the Cauchy-Schwarz inequality,

$$\begin{aligned} f(x) - f(x^*) &\leqslant \langle x - x^*, \nabla f(x) \rangle \\ &\leqslant \|x - x^*\| \|\nabla f(x)\|_* \\ &\leqslant \sqrt{\frac{2}{S}(f(x) - f(x^*))} \|\nabla f(x)\|_*. \end{aligned}$$

Therefore,

$$f(x) - f(x^*) \leqslant \frac{2}{S}\|\nabla f(x^*)\|_*^2. \tag{1.1}$$

If $x = x^*$ then (1.1) is trivially satisfied. Now consider the case of the ℓ_2-norm. By strong convexity, for all $x, y \in \mathbb{R}^n$,

$$f(y) \geqslant f(x) + \langle y - x, \nabla f(x) \rangle + \frac{S}{2}\|y - x\|_2^2$$

With respect to $y \in \mathbb{R}^n$, the left-hand side is minimized for $y = x^*$ and the right-hand side is minimized for $y = x - \nabla f(x)/S$. Thus,

$$\begin{aligned} f(x^*) &\geqslant f(x) + \left\langle -\frac{1}{S}\nabla f(x), \nabla f(x) \right\rangle + \frac{S}{2}\left\| -\frac{1}{S}\nabla f(x) \right\|_2^2 \\ &= f(x) - \frac{\|\nabla f(x)\|_2^2}{2S}, \end{aligned}$$

i.e.,

$$f(x) - \min_{\mathbb{R}^n} f \leqslant \frac{\|\nabla f(x)\|_2^2}{2S}.$$

□

Definition 1.21. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set. A function $f\colon \mathbb{R}^n \to \mathbb{R}$ is σ-sharp on $\mathcal{C}$ if $\sigma > 0$ and for all $x \in \mathcal{C}$,*

$$\operatorname{dist}\left(x, \arg\min_{\mathcal{C}} f\right) \leqslant \sigma\sqrt{f(x) - \min_{\mathcal{C}} f}.$$

Fact 1.22. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $f\colon \mathbb{R}^n \to \mathbb{R}$ be differentiable on $\mathcal{C}$. If f is S-strongly convex on $\mathcal{C}$, then f is $\sqrt{2/S}$-sharp on $\mathcal{C}$.*

Fact 1.23. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $f\colon \mathbb{R}^n \to \mathbb{R}$ be differentiable on $\mathcal{C}$. If f is σ-sharp on $\mathcal{C}$, then f is $1/(2\sigma^2)$-gradient dominated on $\mathcal{C}$.*

1.2.4 Optimization

Definition 1.24. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a convex set and $f\colon \mathbb{R}^n \to \mathbb{R}$ be a function. A point $x^* \in \mathcal{C}$ is a local minimum for f over $\mathcal{C}$ if there exists $\delta > 0$ such that $f(x) \geqslant f(x^*)$ for all $x \in \mathcal{B}(x^*, \delta)$.*

Definition 1.25. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a convex set and $f\colon \mathbb{R}^n \to \mathbb{R}$ be a differentiable function. A point $x^* \in \mathcal{C}$ is stationary for f over $\mathcal{C}$ if for all $x \in \mathcal{C}$, $\langle x - x^*, \nabla f(x^*)\rangle \geqslant 0$.*

Proposition 1.26. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function and $x^* \in \mathcal{C}$. If x^* is a local minimum for f over $\mathcal{C}$, then x^* is a stationary point for f over $\mathcal{C}$.*

CHAPTER 2

THE FRANK-WOLFE ALGORITHM

In this chapter, we conduct an overview of the Frank-Wolfe algorithm, a simple projection-free method for smooth constrained optimization. We explain the basic notions and we discuss the fundamental properties of the algorithm. In the second part, we present convergence rates on convex and nonconvex objectives.

2.1 Description of the algorithm

The Frank-Wolfe algorithm (FW) [43], a.k.a. conditional gradient algorithm [85], is a first-order projection-free algorithm for constrained optimization, addressing

$$\min_{x \in \mathcal{C}} f(x), \tag{2.1}$$

where $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set and $f \colon \mathbb{R}^n \to \mathbb{R}$ is a smooth function. It is presented in Algorithm 2.1.

Algorithm 2.1 Frank-Wolfe (FW)

Input: Start point $x_0 \in \mathcal{C}$, step-size strategy $(\gamma_t)_{t \in \mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$
3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
4: **end for**

A projection-free algorithm. At each iteration, FW solves a linear minimization problem given by the current gradient $\nabla f(x_t)$ (Line 2) and moves in the direction of a solution v_t with a step-size $\gamma_t \in [0, 1]$ (Line 3). This ensures that the new iterate $x_{t+1} = (1 - \gamma_t) x_t + \gamma_t v_t \in \mathcal{C}$ is feasible by convexity. Thus, it does not require a projection back onto $\mathcal{C}$ to ensure feasibility of x_{t+1}. This *projection-free* property is the main purpose of

the algorithm. It is permitted by the access to an oracle computing linear minimizations over $\mathcal{C}$. In the next chapter (Chapter 3), we illustrate the advantage of computing linear minimizations instead of projections for several regions of interest in optimization and machine learning. This property of FW has encountered numerous applications, including solving traffic assignment problems [83], performing video co-localization [66], or, e.g., developing adversarial attacks [20].

Intuition for the algorithm. Suppose we want to design an algorithm ensuring feasibility of its iterates without ever using projections. In this respect, we must use properties of the feasible region $\mathcal{C}$, and we shall leverage all of them, namely, compactness and convexity. Compactness enables us to pick vertices, and convexity guarantees that by moving towards them, we stay feasible. So which vertex to pick at iteration t? Up to here, we have only used information on the feasible region. In order to generate a sequence of iterates converging to a solution, we shall also use properties of the objective function f, namely, smoothness, which gives access to gradient information. Since we are forbidden quadratic programs (else we could as well use projections), we can pick a vertex v_t minimizing the linear approximation of f at x_t over $\mathcal{C}$, i.e.,

$$\min_{v \in \mathcal{C}} f(x_t) + \langle v - x_t, \nabla f(x_t) \rangle.$$

This is exactly

$$\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle,$$

and we have obtained FW. The fact that v_t minimizes the linear approximation if f is indeed a key ingredient in the convergence analysis of FW for convex objectives (Section 2.3) and for defining convergence to a stationary point for nonconvex objectives (Section 2.2).

Step-size strategies. There are two step-size strategies for which convergence of FW has been well studied. The strategy first considered historically [43, 85, 36] was

$$\gamma_t \leftarrow \min\left\{\frac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2}, 1\right\}. \tag{2.2}$$

It is obtained by minimizing the quadratic upper bound from smoothness:

$$\gamma_t = \operatorname*{arg\,min}_{\gamma\in[0,1]} f(x_t) + \gamma\langle v_t - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma^2\|v_t - x_t\|^2, \tag{2.3}$$

and guarantees progress at each iteration, i.e., we have always $f(x_{t+1}) \leqslant f(x_t)$. However, it requires some knowledge of the smoothness constant L of f. To avoid such a requirement, [39] proposed *open loop* strategies, basically in the form $\gamma_t \sim 1/t$. We will refer to

$$\gamma_t \leftarrow \frac{2}{t+2} \tag{2.4}$$

as the *open-loop* strategy, as used in [63]; the strategy (2.2) is thus referred to as the *closed-loop* strategy. The open-loop strategy does not ensure progress at each iteration but it is very simple to implement and its oblivious decaying allows analyses of FW in different settings, e.g., with stochastic gradients.

Sparsity of the iterates. By design, the iterates of FW satisfy $x_t \in \operatorname{conv}\{x_0, v_0, \ldots, v_{t-1}\}$. Thus, if $\mathcal{C}$ is a polytope and x_0 is a vertex, then x_t is the convex combination of at most $t+1$ (unique) vertices. When $\mathcal{C} = \Delta_n$ is the standard simplex, the set of vertices is the standard basis $\{e_1, \ldots, e_n\}$ so FW generates iterates with only a few nonzero entries [21]. When $\mathcal{C} = \{X \in \mathbb{R}^{m\times n} \mid \|X\|_{\text{nuc}} \leqslant 1\}$ is the nuclear norm-ball, the set of vertices are a set of rank-1 matrices $\{uv^\top \mid (u,v) \in \mathbb{R}^m \times \mathbb{R}^n, \|u\|_2 = \|v\|_2 = 1\}$, so FW generates iterates with low rank [56, 64]. This property can also be used to optimize structural sup-

port vector machines (SVMs) [79], estimate Sinkhorn barycenters [92], or, e.g., solve the (approximate) Carathéodory problem [100, 25] (Chapter 6).

Full corrections. We can further improve the sparsity in FW by ensuring that the contribution of each selected vertex is maximized. The Fully-Corrective Frank-Wolfe algorithm (FCFW) [60] computes the new iterate x_{t+1} by reoptimizing f over the convex hull $\operatorname{conv}\{x_0, v_0, \ldots, v_t\}$ of selected vertices. Compared to FW, this avoids selecting redundant vertices in the future. In practice, FCFW generates iterates with much higher sparsity than FW but each iteration is very expensive to compute. FCFW is presented in Algorithm 2.2, where $\mathcal{S}_t$ denotes the set of vertices in the convex decomposition of x_t.

Algorithm 2.2 Fully-Corrective Frank-Wolfe (FCFW)

Input: Start point $x_0 \in \mathcal{C}$.

1: $\mathcal{S}_0 \leftarrow \{x_0\}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$
4: $\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
5: $\quad x_{t+1} \leftarrow \arg\min_{x \in \operatorname{conv} \mathcal{S}_{t+1}} f(x)$
6: **end for**

2.2 The Frank-Wolfe duality gap

The *Frank-Wolfe duality gap* is defined in Definition 2.1. This quantity was introduced in [59]. In the Frank-Wolfe literature, it was first seen in [21] for the standard simplex $\mathcal{C} = \Delta_n$ and was extended to arbitrary sets in [63], which set the name.

Definition 2.1. *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be a differentiable function and $x \in \mathcal{C}$. The Frank-Wolfe duality gap of f at x over $\mathcal{C}$ is $G(x) = \max_{v \in \mathcal{C}} \langle x - v, \nabla f(x) \rangle$.*

The Frank-Wolfe duality gap arises naturally in FW as it is computed at each iteration (Line 2). Without confusion, we will refer to it simply as the *duality gap*. It is a useful quantity in practice as it measures the convergence to a solution when f is convex, and

it is observable even when $\min_{\mathcal{C}} f$ is unknown (Proposition 2.2). In general, it measures the convergence to a stationary point of f over $\mathcal{C}$. Thus, it is often used to analyze the convergence of FW on nonconvex objectives, as first done in [77].

Proposition 2.2. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be a differentiable function and $x \in \mathcal{C}$. Then:*

(i) $G(x) \geqslant 0$,

(ii) $G(x) = 0 \Leftrightarrow x$ *is a stationary point,*

(iii) $f(x) - \min_{\mathcal{C}} f \leqslant G(x)$ *if f is convex.*

Proof. (i) We have

$$\begin{aligned} G(x) &= \max_{v \in \mathcal{C}} \langle x - v, \nabla f(x) \rangle \\ &= \langle x, \nabla f(x) \rangle - \min_{v \in \mathcal{C}} \langle v, \nabla f(x) \rangle \\ &\geqslant 0, \end{aligned}$$

since $x \in \mathcal{C}$.

(ii) We have

$$\begin{aligned} G(x) = 0 &\Leftrightarrow \max_{v \in \mathcal{C}} \langle x - v, \nabla f(x) \rangle = 0 \\ &\Leftrightarrow \forall v \in \mathcal{C}, \langle x - v, \nabla f(x) \rangle \leqslant 0 \\ &\Leftrightarrow x \text{ is a stationary point}, \end{aligned}$$

where in the second equivalence we used that $\max_{v \in \mathcal{C}} \langle x - v, \nabla f(x) \rangle \geqslant 0$ by (i).

(iii) Let $x^* \in \arg\min_{\mathcal{C}} f$. By convexity of f,

$$\begin{aligned} f(x) - \min_{\mathcal{C}} f &= f(x) - f(x^*) \\ &\leqslant \langle x - x^*, \nabla f(x) \rangle \\ &\leqslant \max_{v \in \mathcal{C}} \langle x - v, \nabla f(x) \rangle \\ &= G(x), \end{aligned}$$

since $x^* \in \mathcal{C}$.

□

2.3 Convergence analysis

In this section, $(x_t)_{t \in \mathbb{N}}$ denotes the sequence of iterates generated by FW (Algorithm 2.1).

2.3.1 Key lemmata

We start with a few lemmata.

Lemma 2.3. *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth function. Then for all $t \in \mathbb{N}$,*

$$f(x_{t+1}) \leqslant f(x_t) + \gamma_t \langle v_t - x_t, \nabla f(x_t) \rangle + \frac{L}{2} \gamma_t^2 \|v_t - x_t\|^2.$$

Proof. Let $t \in \mathbb{N}$. We have $x_{t+1} = x_t + \gamma_t(v_t - x_t)$. By L-smoothness of f,

$$\begin{aligned} f(x_{t+1}) &\leqslant f(x_t) + \langle x_{t+1} - x_t, \nabla f(x_t) \rangle + \frac{L}{2} \|x_{t+1} - x_t\|^2 \\ &= f(x_t) + \gamma_t \langle v_t - x_t, \nabla f(x_t) \rangle + \frac{L}{2} \gamma_t^2 \|v_t - x_t\|^2. \end{aligned}$$

□

Lemma 2.4. *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth function and consider the closed-loop strat-*

egy (2.2)*. Then for all* $t \in \mathbb{N}$*,*

$$f(x_{t+1}) \leqslant \begin{cases} f(x_t) - \dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle^2}{2L\|x_t - v_t\|^2} & \text{if } \gamma_t < 1, \\ f(x_t) - \dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle}{2} & \text{if } \gamma_t = 1. \end{cases}$$

Proof. Let $t \in \mathbb{N}$. We have

$$\begin{cases} \gamma_t = \dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2} < 1 & \Leftrightarrow \langle x_t - v_t, \nabla f(x_t)\rangle < L\|v_t - x_t\|^2 \\ \gamma_t = 1 & \Leftrightarrow \langle x_t - v_t, \nabla f(x_t)\rangle \geqslant L\|v_t - x_t\|^2. \end{cases}$$

By Lemma 2.3,

$$\begin{aligned} f(x_{t+1}) &\leqslant f(x_t) + \gamma_t\langle v_t - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma_t^2\|v_t - x_t\|^2 \\ &\leqslant \begin{cases} f(x_t) + \dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|^2}\langle v_t - x_t, \nabla f(x_t)\rangle + \dfrac{L}{2}\dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle^2}{L^2\|x_t - v_t\|^4}\|v_t - x_t\|^2 \\ \qquad\qquad \text{if } \langle x_t - v_t, \nabla f(x_t)\rangle < L\|v_t - x_t\|^2 \\ f(x_t) + \langle v_t - x_t, \nabla f(x_t)\rangle + \dfrac{L}{2}\|v_t - x_t\|^2 \\ \qquad\qquad \text{if } \langle x_t - v_t, \nabla f(x_t)\rangle \geqslant L\|v_t - x_t\|^2 \end{cases} \\ &\leqslant \begin{cases} f(x_t) - \dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle^2}{2L\|x_t - v_t\|^2} & \text{if } \langle x_t - v_t, \nabla f(x_t)\rangle < L\|v_t - x_t\|^2 \\ f(x_t) - \dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle}{2} & \text{if } \langle x_t - v_t, \nabla f(x_t)\rangle \geqslant L\|v_t - x_t\|^2. \end{cases} \end{aligned}$$

□

Lemma 2.5. *Let* $f\colon \mathbb{R}^n \to \mathbb{R}$ *be an* L*-smooth convex function and consider the open-loop strategy* (2.4) *or the closed-loop strategy* (2.2)*. Then*

$$f(x_1) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2}.$$

Proof. Let $x^* \in \arg\min_{\mathcal{C}} f$ be a solution. If the open-loop strategy is used, then $\gamma_0 = 1$ so

by Lemma 2.3, optimality of v_0 (Line 2), and convexity of f,

$$\begin{aligned} f(x_1) &\leqslant f(x_0) + \gamma_0\langle v_0 - x_0, \nabla f(x_0)\rangle + \frac{L}{2}\gamma_0^2\|v_0 - x_0\|^2 \\ &= f(x_0) + \langle v_0 - x_0, \nabla f(x_0)\rangle + \frac{L}{2}\|v_0 - x_0\|^2 \\ &\leqslant f(x_0) + \langle x^* - x_0, \nabla f(x_0)\rangle + \frac{LD^2}{2} \\ &\leqslant f(x^*) + \frac{LD^2}{2}. \end{aligned}$$

If the closed-loop strategy is used, then by Lemma 2.3 and by optimality of the strategy (2.3),

$$\begin{aligned} f(x_1) &\leqslant f(x_0) + \gamma_0\langle v_0 - x_0, \nabla f(x_0)\rangle + \frac{L}{2}\gamma_0^2\|v_0 - x_0\|^2 \\ &= \min_{\gamma\in[0,1]} f(x_0) + \gamma\langle v_0 - x_0, \nabla f(x_0)\rangle + \frac{L}{2}\gamma^2\|v_0 - x_0\|^2 \\ &\leqslant f(x_0) + \langle v_0 - x_0, \nabla f(x_0)\rangle + \frac{L}{2}\|v_0 - x_0\|^2, \end{aligned}$$

and we conclude as before. □

2.3.2 Convergence on convex objectives

First, we consider the case where f is convex. In this situation, FW converges at a rate $\mathcal{O}(LD^2/t)$ (Theorem 2.6). We present the proof techniques from [63] and [43, Sec. 6] for the open- and closed-loop strategies respectively.

Theorem 2.6. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth convex function and consider:*

(i) the open-loop strategy (2.4)*. Then for all $t \geqslant 1$,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{2LD^2}{t+2}.$$

(ii) the closed-loop strategy (2.2)*. Then for all* $t \geqslant 1$*,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{4LD^2}{t+2}.$$

Proof. Let $\varepsilon_t = f(x_t) - \min_{\mathcal{C}} f$ denote the primal gap at iteration $t \in \mathbb{N}$. We proceed by induction.

(i) The base case is satisfied by Lemma 2.5. Suppose that $\varepsilon_t \leqslant 2LD^2/(t+2)$ for some $t \geqslant 1$. Then by Lemma 2.3 and convexity of f,

$$\begin{aligned}
\varepsilon_{t+1} &\leqslant (1-\gamma_t)\varepsilon_t + \frac{LD^2}{2}\gamma_t^2 \\
&\leqslant \frac{t}{t+2}\frac{2LD^2}{t+2} + \frac{2LD^2}{(t+2)^2} \\
&= \frac{2LD^2(t+1)}{(t+2)^2} \\
&\leqslant \frac{2LD^2}{t+3}.
\end{aligned}$$

(ii) The base case is satisfied by Lemma 2.5. Suppose that $\varepsilon_t \leqslant 4LD^2/(t+2)$ for some $t \geqslant 1$. Then by Lemma 2.4 and convexity of f,

$$\begin{aligned}
\varepsilon_{t+1} &\leqslant \begin{cases} \varepsilon_t\left(1 - \dfrac{\varepsilon_t}{2LD^2}\right) & \text{if } \gamma_t < 1 \\ \varepsilon_t/2 & \text{if } \gamma_t = 1 \end{cases} \\
&\leqslant \begin{cases} \dfrac{2LD^2}{t+2} & \text{if } \gamma_t < 1 \text{ and } \varepsilon_t \leqslant \dfrac{2LD^2}{t+2} \\ \dfrac{4LD^2}{t+2}\left(1 - \dfrac{1}{t+2}\right) & \text{if } \gamma_t < 1 \text{ and } \varepsilon_t \geqslant \dfrac{2LD^2}{t+2} \\ \dfrac{2LD^2}{t+2} & \text{if } \gamma_t = 1 \end{cases} \\
&\leqslant \frac{4LD^2}{t+3}.
\end{aligned}$$

□

FW also converges in duality gap (Theorem 2.7) [63].

Theorem 2.7. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth convex function and consider the open-loop strategy* (2.4)*. If $T \geqslant 2$, there exists $t \in [\![\lceil (2/3)T \rceil - 2, T-1]\!]$ such that*

$$G(x_t) \leqslant \frac{(27/4)LD^2}{T-1}.$$

Proof. Let $T \geqslant 2$. We proceed by absurdity. Suppose that for all $t \in [\![\lceil (2/3)T \rceil - 2, T-1]\!]$,

$$G(x_t) > \frac{(27/4)LD^2}{T-1}.$$

Let $\varepsilon_t = f(x_t) - \min_{\mathcal{C}} f$ for all $t \in [\![\lceil (2/3)T \rceil - 2, T]\!]$. By Lemma 2.3, for all $t \in [\![\lceil (2/3)T \rceil - 2, T-1]\!]$,

$$\begin{aligned}
\varepsilon_{t+1} &\leqslant \varepsilon_t - \gamma_t G(x_t) + \frac{LD^2}{2}\gamma_t^2 \\
&\leqslant \varepsilon_t - \gamma_{T-1}\frac{(27/4)LD^2}{T-1} + \frac{LD^2}{2}\gamma^2_{\lceil (2/3)T \rceil - 2} \\
&= \varepsilon_t - \frac{(27/2)LD^2}{T^2-1} + \frac{2LD^2}{\lceil (2/3)T \rceil^2}.
\end{aligned}$$

Let $\Gamma_T = \operatorname{card}[\![\lceil (2/3)T \rceil - 2, T-1]\!]$. By telescoping over $t \in [\![\lceil (2/3)T \rceil - 2, T-1]\!]$ and

with Theorem 2.6,

$$\begin{aligned}
\varepsilon_T &\leqslant \varepsilon_{\lceil (2/3)T \rceil - 2} - \frac{(27/2)LD^2\Gamma_T}{T^2-1} + \frac{2LD^2\Gamma_T}{\lceil (2/3)T \rceil^2} \\
&\leqslant \frac{2LD^2}{\lceil (2/3)T \rceil} - \frac{(27/2)LD^2\Gamma_T}{T^2-1} + \frac{2LD^2\Gamma_T}{\lceil (2/3)T \rceil^2} \\
&\leqslant \frac{2LD^2}{(2/3)T} - \frac{(27/2)LD^2\Gamma_T}{T^2} + \frac{2LD^2\Gamma_T}{(4/9)T^2} \\
&= \frac{(4/3)LD^2T - 6LD^2\Gamma_T + 2LD^2\Gamma_T}{(4/9)T^2} \\
&= \frac{(4/3)LD^2T - 4LD^2\Gamma_T}{(4/9)T^2} \\
&\leqslant \frac{-8LD^2}{(4/9)T^2}
\end{aligned}$$

by using $\Gamma_T = T + 2 - \lceil (2/3)T \rceil \geqslant T/3 + 2$. Therefore,

$$\varepsilon_T < 0.$$

This is absurd. □

The convergence rates of FW in Theorems 2.6–2.7 are tight, as matching lower bounds $\Omega(1/t)$ were established in [63] (Theorem 2.9 and Proposition 2.8). These results were then extended in [80] to provide lower bounds $\Omega(LD^2/t)$ with explicit dependence in the constants L and D. Much earlier, in the 1960s, [17] proved an asymptotic lower bound $\Omega(1/t^{1+\delta})$ for any $\delta > 0$. Note that Proposition 2.8 is actually a lower bound on the number of oracle calls.

Proposition 2.8. *For all* $k \in [\![1, n]\!]$,

$$\min_{\substack{x \in \Delta_n \\ \|x\|_0 \leqslant k}} \|x\|_2^2 = \frac{1}{k}. \tag{2.5}$$

Proof. Let $\Delta_n^{(k)} = \{x \in \Delta_n \mid \|x\|_0 \leqslant k\}$ for all $k \in [\![1, n]\!]$. The base case is trivial since $\Delta_n^{(1)} = \{e_1, \ldots, e_n\}$. Suppose that (2.5) holds for some $k \in [\![1, n-1]\!]$ and let $x \in \Delta_n^{(k+1)}$.

Then there exists $i \in [\![1, n]\!]$ such that $[x]_i \in]0, 1[$. Thus,

$$\begin{cases} \|x - [x]_i e_i\|_0 \leqslant k, \\ \dfrac{x - [x]_i e_i}{1 - [x]_i} \in \Delta_n. \end{cases}$$

It follows that $(x - [x]_i e_i)/(1 - [x]_i) \in \Delta_n^{(k)}$, so

$$\begin{aligned} \|x\|_2^2 &= \|(x - [x]_i e_i) + [x]_i e_i\|_2^2 \\ &= \|(x - [x]_i e_i)\|_2^2 + \|[x]_i e_i\|_2^2 \\ &= (1 - [x]_i)^2 \left\| \frac{x - [x]_i e_i}{1 - [x]_i} \right\|_2^2 + [x]_i^2 \\ &\geqslant (1 - [x]_i)^2 \frac{1}{k} + [x]_i^2 \\ &\geqslant \inf_{\gamma \in]0,1[} (1 - \gamma)^2 \frac{1}{k} + \gamma^2 \\ &= \frac{1}{k+1}. \end{aligned}$$

□

Theorem 2.9. *Let $\mathcal{C} = \Delta_n$ and $f \colon x \in \mathbb{R}^n \mapsto \|x\|_2^2$. Suppose that $x_0 \in \{e_1, \ldots, e_n\}$. Then for all $t \in [\![0, n-1]\!]$,*

$$f(x_t) - \min_{\mathcal{C}} f \geqslant \frac{1}{t+1} - \frac{1}{n},$$

and for all $t \in [\![0, n-2]\!]$,

$$G(x_t) \geqslant \frac{2}{t+1}.$$

Proof. It is implicitly assumed that $T \geqslant n - 1$. First, Proposition 2.8 with $k = n$ shows that $\min_{\mathcal{C}} f = 1/n$. Now, let $t \in [\![0, n-1]\!]$. We have $x_t \in \operatorname{conv}\{x_0, v_0, \ldots, v_{t-1}\}$ so

$\|x_t\|_0 \leqslant t+1$. By Proposition 2.8,

$$\begin{aligned} f(x_t) - \min_{\mathcal{C}} f &= \|x_t\|_2^2 - \frac{1}{n} \\ &\geqslant \frac{1}{t+1} - \frac{1}{n}, \end{aligned}$$

and if $t \leqslant n-2$, then $\min_{i \in [\![1,n]\!]} [x]_i = 0$ so

$$\begin{aligned} G(x_t) &= \max_{i \in [\![1,n]\!]} \langle 2x_t, x_t - e_i \rangle \\ &= 2\|x_t\|_2^2 - 2 \min_{i \in [\![1,n]\!]} [x]_i \\ &\geqslant \frac{2}{t+1}. \end{aligned}$$

□

However, faster rates can be obtained under additional assumptions on the properties of f, the location of the set of unconstrained solutions $\arg\min_{\mathbb{R}^n} f$ relative to $\mathcal{C}$, or the geometry of $\mathcal{C}$ (Section 6.3).

2.3.3 Convergence on nonconvex objectives

Here we do not assume convexity of f. The first analysis of FW on nonconvex objectives probably appeared in [99], without however providing convergence rates. More recently, [77] established a rate $\min_{s \in [\![0,t]\!]} G(x_s) = \mathcal{O}(1/\sqrt{t})$ (Theorem 2.10(i)). Similar bounds holding at a randomly sampled iterate can also be derived (Theorem 2.10(ii)–(iii)); note that in (iii), by setting, e.g., $\nu \leftarrow 0.05$, then $\zeta_{1+2\nu} \approx 10.6$ and $\mathsf{E}[G(X_t)] = \mathcal{O}(1/t^{0.45})$.

Theorem 2.10. *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth nonconvex function and consider:*

(i) the closed-loop strategy (2.2)*. Then for all $t \in \mathbb{N}$,*

$$\min_{s \in [\![0,t]\!]} G(x_s) \leqslant \frac{\max\{LD^2, 2(f(x_0) - \min_{\mathcal{C}} f)\}}{\sqrt{t+1}}.$$

(ii) the fixed-horizon strategy $\gamma_t \leftarrow 1/\sqrt{T}$*. Let* X_{T-1} *be sampled uniformly at random from* $\{x_0, \dots, x_{T-1}\}$*. Then*

$$\mathsf{E}[G(X_{T-1})] \leqslant \frac{(f(x_0) - \min_{\mathcal{C}} f) + LD^2/2}{\sqrt{T}}.$$

(iii) the strategy $\gamma_t \leftarrow 1/(t+1)^{1/2+\nu}$ *for some* $\nu \in \,]0, 1/2[$*. For all* $t \in \mathbb{N}$*, let* X_t *be sampled uniformly at random from* $\{x_0, \dots, x_t\}$*. Then*

$$\mathsf{E}[G(X_t)] \leqslant \frac{(f(x_0) - \min_{\mathcal{C}} f) + LD^2\zeta_{1+2\nu}/2}{(t+1)^{1/2-\nu}},$$

where $\zeta_{1+2\nu} = \sum_{s=0}^{+\infty} 1/(s+1)^{1+2\nu}$.

Proof. (i) Let $t \in \mathbb{N}$. By Lemma 2.4,

$$f(x_t) - f(x_{t+1}) \geqslant \min\left\{\frac{G(x_t)^2}{2LD^2}, \frac{G(x_t)}{2}\right\}$$

so

$$\begin{cases} G(x_t)^2 \leqslant 2LD^2(f(x_t) - f(x_{t+1})) \\ G(x_t) \leqslant 2(f(x_t) - f(x_{t+1})). \end{cases}$$

By telescoping over $s \in [\![0, t]\!]$,

$$\begin{cases} \displaystyle\sum_{s=0}^{t} G(x_s)^2 \leqslant 2LD^2(f(x_0) - f(x_{t+1})) \\ \displaystyle\sum_{s=0}^{t} G(x_s) \leqslant 2(f(x_0) - f(x_{t+1})). \end{cases}$$

Therefore, using $\varepsilon_0 = f(x_0) - \min_{\mathcal{C}} f$,

$$\begin{aligned}\min_{s\in[\![0,t]\!]} G(x_s) &\leqslant \max\left\{\sqrt{\frac{2LD^2\varepsilon_0}{t+1}}, \frac{2\varepsilon_0}{t+1}\right\}\\ &\leqslant \max\left\{\frac{\max\{LD^2, 2\varepsilon_0\}}{\sqrt{t+1}}, \frac{2\varepsilon_0}{t+1}\right\}\\ &= \frac{\max\{LD^2, 2\varepsilon_0\}}{\sqrt{t+1}}.\end{aligned}$$

(ii) Let $t \in \mathbb{N}$. By Lemma 2.3,

$$f(x_{t+1}) \leqslant f(x_t) - \gamma_t G(x_t) + \frac{LD^2}{2}\gamma_t^2.$$

By telescoping over $t \in [\![0, T-1]\!]$,

$$\sum_{t=0}^{T-1} \gamma_t G(x_t) \leqslant f(x_0) - f(x_T) + \frac{LD^2}{2}\sum_{t=0}^{T-1}\gamma_t^2,$$

i.e.,

$$\sum_{t=0}^{T-1} \frac{1}{\sqrt{T}} G(x_t) \leqslant f(x_0) - f(x_T) + \frac{LD^2}{2}.$$

Thus,

$$\mathsf{E}[G(X_{T-1})] = \sum_{t=0}^{T-1}\frac{1}{T}G(x_t) \leqslant \frac{f(x_0) - \min_{\mathcal{C}} f + LD^2/2}{\sqrt{T}}.$$

(iii) Let $t \in \mathbb{N}$. By Lemma 2.3,

$$f(x_{t+1}) \leqslant f(x_t) - \gamma_t G(x_t) + \frac{LD^2}{2}\gamma_t^2.$$

By telescoping over $s \in [\![0, t]\!]$,

$$\begin{aligned}\sum_{s=0}^{t} \gamma_s G(x_s) &\leqslant f(x_0) - f(x_{t+1}) + \frac{LD^2}{2} \sum_{s=0}^{t} \gamma_s^2 \\ &\leqslant f(x_0) - \min_{\mathcal{C}} f + \frac{LD^2 \zeta_{1+2\nu}}{2}.\end{aligned}$$

Thus,

$$\begin{aligned}\mathsf{E}[G(X_t)] &= \sum_{s=0}^{t} \frac{1}{t+1} G(x_s) \\ &\leqslant \frac{1}{(t+1)^{1/2-\nu}} \sum_{s=0}^{t} \gamma_s G(x_s) \\ &\leqslant \frac{f(x_0) - \min_{\mathcal{C}} f + LD^2 \zeta_{1+2\nu}/2}{(t+1)^{1/2-\nu}}.\end{aligned}$$

□

CHAPTER 3

COMPLEXITY OF LINEAR MINIMIZATION AND PROJECTION ON SOME SETS

The Frank-Wolfe algorithm is a method for constrained optimization that relies on linear minimizations, as opposed to projections. Therefore, a motivation put forward in a large body of work on the Frank-Wolfe algorithm is the computational advantage of solving linear minimizations instead of projections. However, the discussions supporting this advantage are often too succinct or incomplete. In this chapter, we review the complexity bounds for both tasks on several sets commonly used in optimization. Projection methods onto the ℓ_p-ball, $p \in]1, 2[\cup]2, +\infty[$, and the Birkhoff polytope are also proposed.

Based on [24].

3.1 Motivation

We consider the constrained optimization problem

$$\min_{x \in \mathcal{C}} f(x), \tag{3.1}$$

where $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set and $f \colon \mathbb{R}^n \to \mathbb{R}$ is a smooth function. Among all general purpose methods addressing problem (3.1), the Frank-Wolfe algorithm has the particularity of never requiring projections onto $\mathcal{C}$. It uses linear minimizations over $\mathcal{C}$ instead and is therefore often referred to as a *projection-free* algorithm in the literature, in the sense that it does not call for solutions to quadratic optimization subproblems.

Thus, a motivation put forward in a large body of work on the Frank-Wolfe algorithm is the computational advantage of solving linear minimizations instead of projections. However, only a few works actually provide examples. On the other hand, the complexities of

linear minimizations over several sets are available in [57, 63, 44], but they do not always (accurately) discuss the complexities of the respective projections. Therefore, while it is intuitive that a linear minimization is simpler to solve than a projection in general, a complete quantitative assessment is necessary to properly motivate the projection-free property of the Frank-Wolfe algorithm.

Contributions. We review the complexity bounds of linear minimizations and projections on several sets commonly used in optimization: the standard simplex, the ℓ_p-balls for $p \in [1, +\infty]$, the nuclear norm-ball, the flow polytope, the Birkhoff polytope, and the permutahedron. These sets are selected because linear minimizations or projections can be solved very efficiently, rather than by resorting to a general purpose method, in which case the analysis is less interesting. We also propose two methods for projecting onto the ℓ_p-ball and the Birkhoff polytope respectively, and we analyze their complexity. Computational experiments for the ℓ_1-ball and the nuclear norm-ball are presented.

We would like to stress that, while it is possible that a projection-based algorithm requires less iterations than the Frank-Wolfe algorithm to find a solution to problem (3.1), our goal here is to demonstrate its advantage in terms of per-iteration complexity.

3.2 Notation and definitions

We work in the Euclidean space $\mathbb{R}^n$ or $\mathbb{R}^{m \times n}$ equipped with the standard scalar product $\langle x, y \rangle = x^\top y$ or $\langle X, Y \rangle = \operatorname{tr}(X^\top Y)$. We denote by $\|\cdot\|$ the norm induced by the scalar product, i.e., the ℓ_2-norm $\|\cdot\|_2$ or the Frobenius norm $\|\cdot\|_{\mathrm{F}}$ respectively.

The signum function is $\operatorname{sign}\colon \lambda \in \mathbb{R} \mapsto 1$ if $\lambda > 0$, -1 if $\lambda < 0$, and 0 if $\lambda = 0$. The characteristic function of an event E is $1_E = 1$ if E is true, else 0. The indicator function of a set $\mathcal{C} \subset \mathbb{R}^n$ is $\iota_{\mathcal{C}}\colon x \in \mathbb{R}^n \mapsto 0$ if $x \in \mathcal{C}$, else $+\infty$. Operations on vectors in $\mathbb{R}^n$, such as $\operatorname{sign}(x), |x|, x^p, \max\{x, y\}, xy$, that are conventionally applied to scalars, are carried out entrywise and return a vector in $\mathbb{R}^n$. The shape of 0 and 1 will be clear from context, i.e., a

scalar or a vector. The identity matrix in $\mathbb{R}^{n\times n}$ is denoted by I_n. The matrix with all ones in $\mathbb{R}^{m\times n}$ is denoted by $J_{m,n}$, and by J_n if $m = n$.

Definition 3.1. *We adopt the real-number infinite-precision model of computation. The complexity of a computational task is the number of arithmetic operations necessary to execute it.*

We ran the experiments on a laptop under Linux Ubuntu 20.04 with Intel Core i7-10750H. The code is available at https://github.com/cyrillewcombettes/complexity.

3.3 Projections versus linear minimizations

The Frank-Wolfe algorithm avoids projections by computing linear minimizations instead. In Table 3.1, we summarize the complexities of a linear minimization and a (Euclidean) projection on several sets commonly used in optimization. That is, we compare the complexities of solving

$$\min_{x\in\mathcal{C}} \langle x, y\rangle \quad \text{and} \quad \min_{x\in\mathcal{C}} \|x - y\|. \tag{3.2}$$

When an exact solution cannot be computed directly, we compare to the complexity of finding an ε-approximate solution; note that the two objectives in (3.2) are homogeneous. In that case, we solve $\min_{x\in\mathcal{C}} \|x - y\|^2$ using any method but the Frank-Wolfe algorithm, since it would go against the purpose of this chapter. Note however that the Frank-Wolfe algorithm can generate a solution with complexity $\mathcal{O}(\mathrm{iter}(\mathcal{C})\,\mathrm{diam}(\mathcal{C})^2/\varepsilon^2)$, where $\mathrm{iter}(\mathcal{C})$ denotes the complexity of an iteration, which amounts to that of a linear minimization over $\mathcal{C}$. When addressing problem (2.1), solving projection subproblems via the Frank-Wolfe algorithm is known as conditional gradient sliding [82].

We now provide details for the complexities reported in Table 3.1. Slightly abusing notation although we may have $\mathrm{card}(\arg\min_{x\in\mathcal{C}}\langle x, y\rangle) > 1$, we write $\arg\min_{x\in\mathcal{C}}\langle x, y\rangle = x^*$ instead of $\arg\min_{x\in\mathcal{C}}\langle x, y\rangle \ni x^*$.

Table 3.1: Complexities of linear minimizations and (Euclidean) projections on some sets commonly used in optimization. We denote by x^* a solution, $\rho = p \sup_{t\in\mathbb{N}} \|x_t\|_{2(p-1)}^{p-1} \|x_t\|_2 < +\infty$ where $(x_t)_{t\in\mathbb{N}}$ is the sequence generated by Algorithm 3.1, by ν and σ_1 the number of nonzero entries and the top singular value of $-Y$ respectively, and by $\varepsilon > 0$ the additive error in the objective of (3.2) when an approximate solution is computed. The constant d_z is defined in (3.12) and $\tilde{\mathcal{O}}$ hides polylogarithmic factors.

Set $\mathcal{C}$	**Linear minimization**	**Projection**
ℓ_p-ball, $p \in \{1, 2, +\infty\}$	$\mathcal{O}(n)$	$\mathcal{O}(n)$
ℓ_p-ball, $p \in]1, 2[\cup]2, +\infty[$	$\mathcal{O}(n)$	$\mathcal{O}(n\rho^2 \|y - x^*\|_2^2/\varepsilon^2)$
Nuclear norm-ball	$\mathcal{O}(\nu \ln(m+n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$	$\mathcal{O}(mn \min\{m, n\})$
Flow polytope	$\mathcal{O}(m+n)$	$\tilde{\mathcal{O}}(m^3 n + n^2)$
Birkhoff polytope	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2 d_z^2/\varepsilon^2)$
Permutahedron	$\mathcal{O}(n \ln(n))$	$\mathcal{O}(n \ln(n) + n)$

3.3.1 The ℓ_1-ball and the standard simplex

Let $\{e_1, \ldots, e_n\}$ denote the standard basis in $\mathbb{R}^n$. The ℓ_1-ball is

$$\{x \in \mathbb{R}^n \mid \|x\|_1 \leqslant 1\} = \operatorname{conv}\{\pm e_1, \ldots, \pm e_n\},$$

and the standard simplex is

$$\Delta_n = \{x \in \mathbb{R}^n \mid \langle x, 1\rangle = 1, x \geqslant 0\} = \operatorname{conv}\{e_1, \ldots, e_n\}.$$

A projection onto the ℓ_1-ball amounts to computing a projection onto the standard simplex, for which the most efficient algorithms have a complexity $\mathcal{O}(n)$; see [28] for a review. On the other hand, linear minimizations are available in closed form: for all $y \in \mathbb{R}^n$,

$$\arg\min_{x\in\Delta_n} \langle x, y\rangle = e_{i_{\min}} \quad \text{and} \quad \arg\min_{\|x\|_1\leqslant 1} \langle x, y\rangle = -\operatorname{sign}([y]_{i_{\max}}) e_{i_{\max}},$$

where $i_{\min} \in \arg\min_{i\in[\![1,n]\!]} [y]_i$ and $i_{\max} \in \arg\max_{i\in[\![1,n]\!]} |[y]_i|$. Thus, while their complexities can both be written $\mathcal{O}(n)$, in practice linear minimizations are much simpler to solve than projections. Figure 3.1 presents a computational comparison. The results are averaged

over 5 runs and the shaded areas represent ± 1 standard deviation.

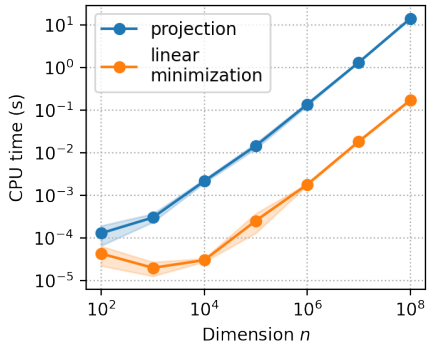

Figure 3.1: Solving a linear minimization and a projection on the ℓ_1-ball. The input vector $y \in \mathbb{R}^n$ is generated by sampling entries from the standard normal distribution and the projection method is [28, Fig. 2], which is state-of-the-art in practice [28, Tab. 3]. In this situation, the plots suggest that linear minimizations are about $100\times$ faster to solve than projections when n is large enough. Exact CPU times are reported in Table B.1 in Appendix B.1.

3.3.2 The ℓ_2-ball and the ℓ_∞-ball

Here, linear minimizations have no significant advantage over projections as they are all available in closed form. For all $y \in \mathbb{R}^n$,

$$\arg\min_{\|x\|_2 \leqslant 1} \langle x, y \rangle = -\frac{y}{\|y\|_2} \quad \text{and} \quad \arg\min_{\|x\|_2 \leqslant 1} \|x - y\|_2 = \frac{y}{\max\{\|y\|_2, 1\}},$$

and

$$\arg\min_{\|x\|_\infty \leqslant 1} \langle x, y \rangle = -\operatorname{sign}(y) \quad \text{and} \quad \arg\min_{\|x\|_\infty \leqslant 1} \|x - y\|_2 = \operatorname{sign}(y) \min\{|y|, 1\}.$$

3.3.3 The ℓ_p-balls for $p \in]1, +\infty[$

Let $p \in]1, +\infty[$. The ℓ_p-ball is $\{x \in \mathbb{R}^n \mid \|x\|_p \leqslant 1\}$. Linear minimizations are available in closed form: by duality, for all $y \in \mathbb{R}^n$,

$$\arg\min_{\|x\|_p \leqslant 1} \langle x, y \rangle = -\nabla \|\cdot\|_q(y) = -\frac{\operatorname{sign}(y)|y|^{q-1}}{\|y\|_q^{q-1}},$$

where $q = p/(p-1) \in]1, +\infty[$. To the best of our knowledge, there is no projection method specific to the ℓ_p-ball when $p \in]1, 2[\cup]2, +\infty[$. We use [27, Alg. 6.5], a Haugazeau-like algorithm [55] for projecting onto the intersection of lower level sets of convex functions. For a single lower level set, the problem reads

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} \|x - y\|_2^2 \\ &\text{s.t. } g(x) \leqslant 0, \end{aligned} \tag{3.3}$$

and we assume that $g \colon \mathbb{R}^n \to \mathbb{R}$ is convex and differentiable for ease of exposition. In our case, $g = \|\cdot\|_p^p - 1$. Alternatively, one could use a Lagrange multiplier to formulate (3.3) as a strongly convex unconstrained problem, but finding the corresponding multiplier may require a considerable effort of tuning; also note that information is usually given in the form $g(x) \leqslant 0$ rather than in the form of a Lagrange multiplier. The method is presented in Algorithm 3.1, where for all $a, b \in \mathbb{R}^n$,

$$H(a, b) = \{x \in \mathbb{R}^n \mid \langle x - b, a - b \rangle \leqslant 0\},$$

and G_t is the projection of x_t onto $\{x \in \mathbb{R}^n \mid g(x_t) + \langle x - x_t, \nabla g(x_t) \rangle \leqslant 0\} = H(x_t, G_t)$. If $g(x_t) \leqslant 0$, then $x_{t+s} = x^*$ for all $s \in \mathbb{N}$ [27, Prop. 3.1], where x^* is the solution to problem (3.3).

Algorithm 3.1 Haugazeau-like for problem (3.3)

Input: Point to project $y \in \mathbb{R}^n$.

1: $x_0 \leftarrow y$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad G_t \leftarrow x_t - \dfrac{g(x_t)}{\|\nabla g(x_t)\|_2^2} \nabla g(x_t) 1_{\{g(x_t) > 0\}}$
4: $\quad x_{t+1} \leftarrow \mathrm{proj}(x_0, H(x_0, x_t) \cap H(x_t, G_t))$
5: **end for**

The projection in Line 4 is available in closed form [55, Thm. 3-1]; see also [6, Cor. 29.25]. The complexity of an iteration of Algorithm 3.1 is $\mathcal{O}(n)$. We propose in Theorem 3.2 the

convergence rate of Algorithm 3.1, based on a key result from [5, Thm. 7.12]. A convergence rate is also proposed in [109], however it uses a stronger assumption and has a minor error in the exponent of the constant.

Theorem 3.2. *Let $g\colon \mathbb{R}^n \to \mathbb{R}$ be a differentiable convex function and $\mathcal{C} = \{x \in \mathbb{R}^n \mid g(x) \leqslant 0\}$, and suppose that there exists $\hat{x} \in \mathcal{C}$ such that $g(\hat{x}) < 0$. Consider Algorithm 3.1 and let $x^* = \operatorname{proj}(x_0, \mathcal{C})$. Then, for all $t \in \mathbb{N}$,*

$$\|x^* - x_0\|_2^2 - \|x_t - x_0\|_2^2 \leqslant \frac{\max\{8\rho^2, 2\}\|x_0 - x^*\|_2^2}{t+2}, \tag{3.4}$$

and

$$\|x_t - x^*\|_2 \leqslant \frac{\max\{2\sqrt{2}\rho, \sqrt{2}\}\|x_0 - x^*\|_2}{\sqrt{t+2}}, \tag{3.5}$$

where $\rho = (-1/g(\hat{x}))\sup_{t\in\mathbb{N}} \|\nabla g(x_t)\|_2\|x_t - \hat{x}\|_2 < +\infty$.

Proof. First, note that by [27, Prop. 3.1], for all $t \in \mathbb{N}$,

$$\|x_t - x_0\|_2 \leqslant \|x_{t+1} - x_0\|_2 \leqslant \|x^* - x_0\|_2. \tag{3.6}$$

We prove by induction that (3.4) holds for all $t \in \mathbb{N}$. The base case $t = 0$ is trivial. Suppose that (3.4) holds at iteration $t \in \mathbb{N}$. Since $x_{t+1} \in H(x_0, x_t)$ and $x_{t+1} \in H(x_t, G_t)$, we have

$$\begin{aligned}\|x_{t+1} - x_0\|_2^2 - \|x_t - x_0\|_2^2 &= \|x_{t+1} - x_t\|_2^2 + 2\langle x_{t+1} - x_t, x_t - x_0\rangle \\ &\geqslant \|x_{t+1} - x_t\|_2^2 \\ &\geqslant \operatorname{dist}(x_t, H(x_t, G_t))^2. \end{aligned} \tag{3.7}$$

By [5, Thm. 7.12],

$$\operatorname{dist}(x_t, \mathcal{C}) \leqslant \rho \operatorname{dist}(x_t, H(x_t, G_t)), \tag{3.8}$$

where $\rho = (-1/g(\hat{x}))\sup_{t\in\mathbb{N}} \|\nabla g(x_t)\|_2\|x_t - \hat{x}\|_2$, and $\rho < +\infty$ because $(x_t)_{t\in\mathbb{N}}$ converges [6, Cor. 30.9]. Now, $x^* = \operatorname{proj}(x_0, \mathcal{C})$ so $\mathcal{C} \subset H(x_0, x^*)$. We can assume that $x_t \neq x^*$, so $x_t \notin H(x_0, x^*)$ by (3.6). By [6, Ex. 29.20],

$$\operatorname{dist}(x_t, H(x_0, x^*)) = \frac{\langle x_t - x^*, x_0 - x^*\rangle}{\|x_0 - x^*\|_2}.$$

Thus,

$$\begin{aligned}
\operatorname{dist}(x_t, \mathcal{C}) &\geqslant \operatorname{dist}(x_t, H(x_0, x^*)) \\
&= \frac{\langle x_t - x^*, x_0 - x^*\rangle}{\|x_0 - x^*\|_2} \\
&= \frac{\langle x_t - x_0, x_0 - x^*\rangle + \|x_0 - x^*\|_2^2}{\|x_0 - x^*\|_2} \\
&\geqslant \|x_0 - x^*\|_2 - \|x_t - x_0\|_2 \\
&= \|x_0 - x^*\|_2 - \sqrt{\|x^* - x_0\|_2^2 - (\|x^* - x_0\|_2^2 - \|x_t - x_0\|_2^2)} \qquad (3.9) \\
&\geqslant 0,
\end{aligned}$$

where we used the Cauchy-Schwarz inequality in the second inequality and (3.6) in the last inequality. Let $\varepsilon_s = \|x^* - x_0\|_2^2 - \|x_s - x_0\|_2^2$ for $s \in \{t, t+1\}$. Combining (3.7)–(3.9),

$$\varepsilon_{t+1} \leqslant \varepsilon_t - \frac{1}{\rho^2}\left(\|x_0 - x^*\|_2 - \sqrt{\|x^* - x_0\|_2^2 - \varepsilon_t}\right)^2.$$

Since $\sqrt{\alpha - \beta} \leqslant \sqrt{\alpha} - \beta/(2\sqrt{\alpha})$ for all $\alpha \geqslant \beta > 0$, we obtain

$$\begin{aligned}
\varepsilon_{t+1} &\leqslant \varepsilon_t - \frac{1}{\rho^2}\left(\|x_0 - x^*\|_2 - \left(\|x^* - x_0\|_2 - \frac{\varepsilon_t}{2\|x^* - x_0\|_2}\right)\right)^2 \\
&= \left(1 - \frac{\varepsilon_t}{4\rho^2\|x^* - x_0\|_2^2}\right)\varepsilon_t.
\end{aligned}$$

Let $\kappa = \max\{8\rho^2, 2\}\|x_0 - x^*\|_2^2$. We have $\varepsilon_t \leqslant \kappa/(t+2)$ by the induction hypothesis, so

$$\begin{aligned}
\varepsilon_{t+1} &\leqslant \left(1 - \frac{\varepsilon_t}{\kappa/2}\right)\varepsilon_t \\
&\leqslant \begin{cases} \dfrac{\kappa/2}{t+2} & \text{if } \varepsilon_t \leqslant \dfrac{\kappa/2}{t+2} \\ \left(1 - \dfrac{1}{t+2}\right)\dfrac{\kappa}{t+2} & \text{if } \varepsilon_t \geqslant \dfrac{\kappa/2}{t+2} \end{cases} \\
&\leqslant \frac{\kappa}{t+3}.
\end{aligned}$$

We conclude that (3.4) holds for all $t \in \mathbb{N}$. Then, for all $t \in \mathbb{N}$,

$$\begin{aligned}
\|x_t - x^*\|_2^2 &= \|x^* - x_0\|_2^2 - \|x_t - x_0\|_2^2 + 2\langle x^* - x_t, x_0 - x_t\rangle \\
&\leqslant \|x^* - x_0\|_2^2 - \|x_t - x_0\|_2^2,
\end{aligned}$$

because $x^* \in H(x_0, x_t)$, since $\mathcal{C} \subset H(x_0, x_t)$ [27, Prop. 5.2]. This proves (3.5). □

In our case, $g = \|\cdot\|_p^p - 1$ and $g(0) = -1 < 0$, so Theorem 3.2 holds with

$$\rho = \sup_{t\in\mathbb{N}} \|\nabla g(x_t)\|_2 \|x_t\|_2 = p \sup_{t\in\mathbb{N}} \|x_t\|_{2(p-1)}^{p-1} \|x_t\|_2 < +\infty.$$

Therefore, the complexity of an ε-approximate projection onto the ℓ_p-ball is $\mathcal{O}(n\rho^2\|y - x^*\|_2^2/\varepsilon^2)$.

Remark 3.3. *If $p \in [1, +\infty[\cap \mathbb{Q}$, another option, although probably less practical, is to formulate the projection problem as a conic quadratic program and to obtain an ε-approximate solution using an interior-point algorithm, with complexity $\mathcal{O}(\mathrm{poly}(n)\ln(1/\varepsilon))$ [8].*

3.3.4 The nuclear norm-ball

This is probably the most popular example of the computational advantage of linear minimizations over projections in the literature. The nuclear norm, a.k.a. trace norm, of a matrix

is the sum of its singular values and serves as a convex surrogate for the rank constraint [42]. The nuclear norm-ball is the convex hull of rank-1 matrices:

$$\{X \in \mathbb{R}^{m\times n} \mid \|X\|_{\text{nuc}} \leqslant 1\} = \operatorname{conv}\{uv^\top \mid u \in \mathbb{R}^m, v \in \mathbb{R}^n, \|u\|_2 = \|v\|_2 = 1\}.$$

For all $Y \in \mathbb{R}^{m\times n}$,

$$\operatorname*{arg\,min}_{\|X\|_{\text{nuc}}\leqslant 1} \|X - Y\|_{\text{F}} = U \operatorname{diag}(\hat{\sigma}) V^\top,$$

where $Y = U \operatorname{diag}(\sigma) V^\top$ is the singular value decomposition (SVD) of Y, $(U, \sigma, V) \in \mathbb{R}^{m\times k} \times \mathbb{R}^k \times \mathbb{R}^{n\times k}$, $k = \min\{m, n\}$, and $\hat{\sigma}$ is the projection of σ onto the standard simplex Δ_k. The SVD can be computed with complexity $\mathcal{O}(mn \min\{m, n\} + \min\{m^3, n^3\})$ using the Golub-Reinsch algorithm or the R-SVD algorithm [49, Fig. 8.6.1]. On the other hand, a linear minimization requires only a truncated SVD:

$$\operatorname*{arg\,min}_{\|X\|_{\text{nuc}}\leqslant 1} \langle X, Y\rangle = \operatorname*{arg\,max}_{\|u\|_2=\|v\|_2=1} \operatorname{tr}((uv^\top)^\top(-Y)) = \operatorname*{arg\,max}_{\|u\|_2=\|v\|_2=1} u^\top(-Y)v = uv^\top,$$

where u and v are the top left and right singular vectors of $-Y$. A pair of unit vectors $(u, v) \in \mathbb{R}^m \times \mathbb{R}^n$ satisfying $\sigma_1 - u^\top(-Y)v \leqslant \varepsilon$ with high probability can be obtained using the Lanczos algorithm with complexity $\mathcal{O}(\nu \ln(m + n)\sqrt{\sigma_1}/\sqrt{\varepsilon})$, where σ_1 and ν denote the top singular value and the number of nonzero entries in $-Y$ respectively [63, 74]. Note that $\nu \leqslant mn$ and that in many applications of interest, e.g., in recommender systems, $\nu \ll mn$.

In practice, the package ARPACK [84] is often used to compute the top pair of singular vectors. Furthermore, if the input matrix Y is symmetric, then the package LOBPCG [72] can be particularly efficient. Figure 3.2 illustrates both cases, where linear minimizations are solved to machine precision. The results are averaged over 5 runs and the shaded areas represent ± 1 standard deviation.

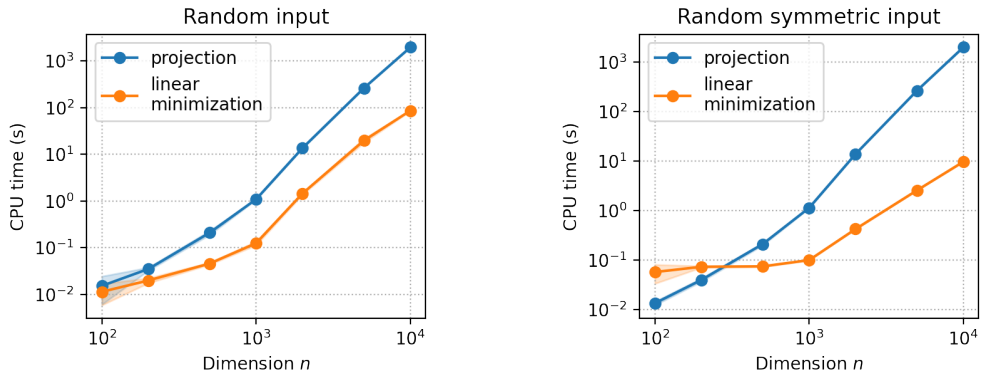

Figure 3.2: Solving a linear minimization and a projection on the nuclear norm-ball. A matrix $Y \in \mathbb{R}^{n\times n}$ is generated by sampling entries from the standard normal distribution. The full and truncated SVDs are computed using the functions `svd` and `svds` from the Python packages `numpy.linalg` [54] and `scipy.sparse.linalg` [128] respectively. The function `svds` is used with `tol=0`. *Left:* The input is Y and the function `svds` is used with `solver='arpack'`. *Right:* The input is the symmetric matrix $(Y + Y^\top)/2$ and the function `svds` is used with `solver='lobpcg'`. We see that the ratio of CPU times increases as n increases. Exact CPU times are reported in Tables B.2–B.3 in Appendix B.1.

3.3.5 The flow polytope

Let G be a single-source single-sink directed acyclic graph (DAG) with m vertices and n edges. Index by $[\![1, m]\!]$ the set of vertices such that the edges are directed from a smaller to a larger vertex index; this can be achieved with complexity $\mathcal{O}(m + n)$ via topological sort [29, Sec. 22.4]. Let $A_G \in \mathbb{R}^{m\times n}$ be the incidence matrix of G. The flow polytope induced by G is

$$\mathcal{F}_G = \{x \in \mathbb{R}^n \mid A_G x = (-1, 0, \dots, 0, 1)^\top, x \geqslant 0\},$$

i.e., $\mathcal{F}_G$ is the set of unit flows $x \in \mathbb{R}^n$ on G, where $[x]_i \geqslant 0$ denotes the flow going through edge i. Thus, for all $y \in \mathbb{R}^n$, $\arg\min_{x\in\mathcal{F}_G} \langle x, y\rangle$ is a flow $x \in \{0, 1\}^n$ identifying a shortest path on G weighted by y. Its computation has complexity $\mathcal{O}(m + n)$ [29, Sec. 24.2]. This is significantly cheaper than the complexity $\tilde{\mathcal{O}}(m^3 n + n^2)$ of a projection [127, Thm. 20], where $\tilde{\mathcal{O}}$ hides polylogarithmic factors.

3.3.6 The Birkhoff polytope

The Birkhoff polytope, a.k.a. assignment polytope, is the set of doubly stochastic matrices

$$\mathcal{B}_n = \{X \in \mathbb{R}^{n\times n} \mid X1 = 1, X^\top 1 = 1, X \geqslant 0\}.$$

It is the convex hull of the permutation matrices and arises in matching, ranking, and seriation problems. Linear minimizations can be solved with complexity $\mathcal{O}(n^3)$ using the Hungarian algorithm [75]. To the best of our knowledge, there is no projection method specific to the Birkhoff polytope so we propose one here. Let $Y \in \mathbb{R}^{n\times n}$. By reshaping it into a vector $y \in \mathbb{R}^{n^2}$, projecting Y onto the Birkhoff polytope is equivalent to solving

$$\begin{aligned} \min_{x\in\mathbb{R}^{n^2}} \ & \|x - y\|_2^2 \\ \text{s.t. } & Ax = 1 \\ & x \geqslant 0, \end{aligned} \qquad \text{where} \quad A = \begin{pmatrix} 1^\top & 0^\top & \cdots & 0^\top \\ 0^\top & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0^\top \\ 0^\top & \cdots & 0^\top & 1^\top \\ I_n & \cdots & \cdots & I_n \end{pmatrix} \in \mathbb{R}^{2n\times n^2}. \tag{3.10}$$

This can again be reformulated as

$$\min_{x\in\mathbb{R}^{n^2}} \left(\iota_{\mathcal{K}}(x) + \frac{1}{2}\|x - y\|_2^2\right) + \left(\iota_{\mathcal{A}}(x) + \frac{1}{2}\|x - y\|_2^2\right), \tag{3.11}$$

where $\mathcal{K} = \{x \in \mathbb{R}^{n^2} \mid x \geqslant 0\}$ and $\mathcal{A} = \{x \in \mathbb{R}^{n^2} \mid Ax = 1\}$. That is, we split the constraints into two sets enjoying efficient projections. We can now apply the Douglas-Rachford algorithm [87] to problem (3.11). The method is presented in Algorithm 3.2; see Appendix A.2 for details. Line 3 computes the projection of u_t onto the affine subspace $\mathcal{A}$ and in Line 4 is computed a projection onto the nonnegative orthant $\mathcal{K}$. We can set $u = 1/n \in \mathcal{A}$ and we denote by $A^\dagger \in \mathbb{R}^{n^2\times 2n}$ the Moore-Penrose inverse of A.

The complexity of an iteration of Algorithm 3.2 is dominated by the matrix-vector

Algorithm 3.2 Douglas-Rachford for problem (3.11)

Input: Point to project $y \in \mathbb{R}^{n^2}$, start point $z_0 \in \mathbb{R}^{n^2}$, offset point $u \in \mathcal{A}$.

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad u_t \leftarrow \dfrac{z_t + y}{2}$

3: $\quad x_t \leftarrow u_t - A^\dagger A(u_t - u)$

4: $\quad z_{t+1} \leftarrow \max\left\{\dfrac{2x_t - z_t + y}{2}, 0\right\} + z_t - x_t$

5: **end for**

multiplication in Line 3. We can assume that $A^\dagger A \in \mathbb{R}^{n^2 \times n^2}$ is precomputed. In fact,

$$A^\dagger A = \frac{1}{n^2}\begin{pmatrix} B_1 & B_2 & \cdots & B_2 \\ B_2 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & B_2 \\ B_2 & \cdots & B_2 & B_1 \end{pmatrix} \in \mathbb{R}^{n^2 \times n^2}, \quad \text{where} \quad \begin{cases} B_1 = nI_n + (n-1)J_n \in \mathbb{R}^{n \times n} \\ B_2 = nI_n - J_n \in \mathbb{R}^{n \times n}, \end{cases}$$

so $A^\dagger A$ is block circulant with circulant blocks (BCCB) and has only three distinct entries: $2n-1$, $n-1$, and -1. The expression of $A^\dagger A$ can be shown by checking that $A^\dagger = A^\top/n - J_{n^2,2n}/(2n^2) \in \mathbb{R}^{n^2 \times 2n}$ using the necessary and sufficient Moore-Penrose conditions [112, Thm. 1]. Thus, the multiplication of $A^\dagger A$ and any vector $x \in \mathbb{R}^{n^2}$ can be performed with complexity $\mathcal{O}(n^2)$. Indeed, it amounts to computing $(nI_n + (n-1)J_n)[x]_{i:j}$ and $(nI_n - J_n)[x]_{i:j}$ for every $(i, j) \in \{(kn+1, (k+1)n) \mid k \in [\![0, n-1]\!]\}$, each of which has complexity $\mathcal{O}(n)$.

It remains to bound the number of iterations required to achieve ε-convergence. Let $x^* = \mathrm{proj}(y, \mathcal{A} \cap \mathcal{K})$ be the solution to problem (3.11), i.e., the projection of Y onto the Birkhoff polytope after reshaping, and let $\bar{x}_t = (\sum_{s=0}^t x_s)/(t+1) \in \mathcal{A}$ for all $t \in \mathbb{N}$. By [32, Thm. 1],

$$\|\bar{x}_t - x^*\|_2 \leqslant \frac{\|z_0 - z^*\|_2}{\sqrt{2(t+1)}},$$

where z^* is a fixed point of $\mathrm{rprox}_{\iota_\mathcal{K} + (1/2)\|\cdot - y\|_2^2} \circ \mathrm{rprox}_{\iota_\mathcal{A} + (1/2)\|\cdot - y\|_2^2}$, $\mathrm{rprox}_\varphi = 2\,\mathrm{prox}_\varphi - \mathrm{id}$,

and prox_{φ} is the proximity operator of φ [101]. Therefore, the complexity of an ε-approximate projection onto the Birkhoff polytope is $\mathcal{O}(n^2 d_z^2/\varepsilon^2)$, where

$$d_z = \|z_0 - z^*\|_2. \tag{3.12}$$

3.3.7 The permutahedron

Let $\mathfrak{S}_n$ be the set of permutations on $[\![1, n]\!]$ and $w \in \mathbb{R}^n$. The permutahedron induced by w is the convex hull of all permutations of the entries in w, i.e.,

$$\mathcal{P}_w = \mathrm{conv}\{w_\sigma \in \mathbb{R}^n \mid w_\sigma = ([w]_{\sigma_1}, \ldots, [w]_{\sigma_n})^\top, \sigma \in \mathfrak{S}_n\}.$$

It is related to the Birkhoff polytope via $\mathcal{P}_w = \{Xw \mid X \in \mathcal{B}_n\}$. With no loss of generality, we can assume that the weights are already sorted in ascending order: $[w]_1 \leqslant \cdots \leqslant [w]_n$. Thus, for all $y \in \mathbb{R}^n$,

$$\underset{x \in \mathcal{P}_w}{\arg\min} \langle x, y \rangle = w_{\sigma^{-1}},$$

where σ satisfies $[y]_{\sigma_1} \geqslant \cdots \geqslant [y]_{\sigma_n}$. Sorting the entries of y has complexity $\mathcal{O}(n \ln(n))$ [29]. A projection can be obtained with a slightly higher complexity $\mathcal{O}(n \ln(n) + n)$, by sorting the entries of y and solving an isotonic regression problem [106].

CHAPTER 4

BOOSTING FRANK-WOLFE BY CHASING GRADIENTS

The main drawback of the Frank-Wolfe algorithm lies in its convergence rate, which can be excessively slow due to naive descent directions. We propose to speed up the Frank-Wolfe algorithm by better aligning the descent direction with that of the negative gradient via a subroutine. This subroutine chases the negative gradient direction in a matching pursuit-style while still preserving the projection-free property. Although the approach is reasonably natural, it produces very significant results. We derive convergence rates $\mathcal{O}(1/t)$ to $\mathcal{O}(e^{-\omega t})$ of our method and we demonstrate its competitive advantage both per iteration and in CPU time over the state of the art in a series of computational experiments.

Based on [23].

4.1 The Frank-Wolfe zig-zagging phenomenon

We address the constrained convex optimization problem

$$\min_{x \in \mathcal{C}} f(x), \tag{4.1}$$

where $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set and $f \colon \mathbb{R}^n \to \mathbb{R}$ is a smooth convex function. In this setting, the Frank-Wolfe algorithm (FW) converges at a rate $\mathcal{O}(1/t)$ (Theorem 2.6). Furthermore, this result is exact, i.e., there exists an instance of problem (4.1) where FW converges at a rate $\Omega(1/t)$ (Theorem 2.9).

FW can converge at faster rates, as we will see in Chapter 6, but this requires additional assumptions on the geometry of $\mathcal{C}$ or the position of the set of unconstrained solutions $\arg\min_{\mathbb{R}^n} f$ relative to $\mathcal{C}$. In particular, further assumptions on the properties of the objective f, e.g., strong convexity, are not sufficient. If $\mathcal{C}$ is a polytope, then the set of

unconstrained solutions needs to lie in the (relative) interior of $\mathcal{C}$ to guarantee faster rates [50], i.e., the constraints need to be not active. This is not an interesting situation in practice, as constraints are usually added to enforce some desired property on the solution, e.g., sparsity, regularization, or fairness, that would not be satisfied otherwise.

Thus, the behavior of FW is very different to that of the projected gradient descent algorithm, which converges at a linear rate $\mathcal{O}(\exp(-(S/L)t))$ when the objective is L-smooth and S-strongly convex, with no additional assumption on the geometry of $\mathcal{C}$. The reason for this is that FW is allowed to move only towards vertices returned by the linear minimization oracle, which can induce a very naive and inefficient zig-zagging trajectory of the iterates. Figure 4.1 illustrates this zig-zagging phenomenon, where the descent directions $v_t - x_t$ become more and more orthogonal to the optimal directions $x^* - x_t$.

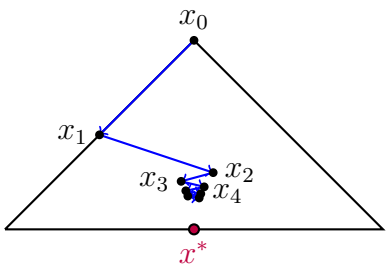

Figure 4.1: FW generates an inefficient zig-zagging trajectory towards the solution. The problem is to minimize $(1/2)\|\cdot\|_2^2$ over the triangular region $\mathrm{conv}\{(-1,0)^\top,(1,0)^\top,(0,1)^\top\}$. The solution is $x^* = (0,0)^\top$ and the start point is $x_0 = (0,1)^\top$.

Contributions. We propose the *Boosted Frank-Wolfe* algorithm (BoostFW), an intuitive method speeding up the Frank-Wolfe algorithm by chasing the negative gradient direction $-\nabla f(x_t)$ via a matching pursuit-style subroutine, and moving in this better aligned direction. BoostFW thereby mimics gradient descent while remaining projection-free. We derive convergence rates $\mathcal{O}(1/t)$ to $\mathcal{O}(e^{-\omega t})$. Although the linear minimization oracle may be called multiple times per iteration, we demonstrate in a series of computational experiments the competitive advantage both per iteration and in CPU time of our method over the state of the art. Furthermore, BoostFW does not require line search to achieve strong

empirical performance, and it does not need to maintain the decomposition of the iterates. Naturally, our approach can also be used to boost the performance of any Frank-Wolfe-style algorithm.

Outline. We start by presenting faster variants of the Frank-Wolfe algorithm (Section 4.2). We then move on to the intuition behind the design of the Boosted Frank-Wolfe algorithm and present its convergence analysis (Section 4.3). Applications of the boosting procedure to other algorithms are discussed in Section 4.3.4. We validate the advantage of our method in a series of computational experiments (Section 4.4). Finally, a couple of remarks conclude the chapter (Section 4.5). Complementary plots can be found in Appendix B.2.

4.2 Faster variants of the Frank-Wolfe algorithm

In this section, we present some core variants of FW. They guarantee linear convergence rates on L-smooth S-strongly convex objectives over polytopes without excessively increasing the per-iteration complexity, and work well in practice.

Away steps. To solve the zig-zagging issue in FW, [131] proposed the Away-Step Frank-Wolfe algorithm (AFW, Algorithm 4.1), which was analyzed in [50] and recently reanalyzed in [78]. The idea is to allow FW to move also away from vertices. At each iteration, AFW calls the linear minimization oracle twice to return a Frank-Wolfe vertex and an *away* vertex and compares which direction will provide more progress. Thus, it is able to notice when zig-zagging starts and converges much faster overall (Figure 4.2). Its rate of convergence is $\mathcal{O}(\exp(-(S/(8L))(W/D)^2 t))$, where W is the *pyramidal width* of the polytope [78]. For example, for the standard simplex Δ_n, $(W/D)^2 = 2/n$ when n is even and $(W/D)^2 = 2/(n - 1/n)$ when n is odd.

Memory usage. In order to ensure feasibility of its iterates when performing away steps, AFW needs to store their convex decomposition onto the vertices of $\mathcal{C}$. This can be-

Algorithm 4.1 Away-Step Frank-Wolfe (AFW)

Input: Start point $x_0 \in \mathcal{C}$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0,1]$.

1: $\mathcal{S}_0 \leftarrow \{x_0\}$
2: $\lambda_0(x_0) \leftarrow 1$
3: **for** $t = 0$ **to** $T-1$ **do**
4: $\quad v_t \leftarrow \arg\min_{v\in\mathcal{C}} \langle v, \nabla f(x_t)\rangle$ ▷ FW vertex
5: $\quad a_t \leftarrow \arg\max_{v\in\mathcal{S}_t} \langle v, \nabla f(x_t)\rangle$ ▷ away vertex
6: $\quad$ **if** $\langle x_t - v_t, \nabla f(x_t)\rangle \geqslant \langle a_t - x_t, \nabla f(x_t)\rangle$ **then**
7: $\quad\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$ ▷ FW step
8: $\quad\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
9: $\quad\quad \lambda_{t+1}(v) \leftarrow (1-\gamma_t)\lambda_t(v) + \gamma_t 1_{\{v=v_t\}}$ **for** $v \in \mathcal{S}_{t+1}$
10: $\quad$ **else**
11: $\quad\quad \gamma_{\max} \leftarrow \lambda_t(a_t)/(1-\lambda_t(a_t))$
12: $\quad\quad x_{t+1} \leftarrow x_t + \gamma_t(x_t - a_t)$ where $\gamma_t \leqslant \gamma_{\max}$ ▷ away step
13: $\quad\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t$
14: $\quad\quad \lambda_{t+1}(v) \leftarrow (1+\gamma_t)\lambda_t(v) - \gamma_t 1_{\{v=a_t\}}$ **for** $v \in \mathcal{S}_{t+1}$
15: $\quad$ **end if**
16: **end for**

come very expensive in memory usage. The Decomposition-Invariant Pairwise Conditional Gradient algorithm (DICG, Algorithm 4.2) [46] is a variant over $0/1$-polytopes that is memory-free by leveraging the structure of these polytopes. DICG converges at a rate $\mathcal{O}(\exp(-(S/(8LD^2\|x^*\|_0))t))$, where $\|x^*\|_0$ denotes the number of nonzero entries in the solution $x^* = \arg\min_{\mathcal{C}} f$. Thus, comparing to AFW, the square of the pyramidal width W^2, which is often a dimension-dependent quantity, is replaced with the sparsity of x^*, which can be much smaller. Note that an extension of DICG to arbitrary polytopes was proposed in [4], however it assumes access to an expensive oracle hindering its performance in practice.

Blending. The Blended Conditional Gradient algorithm (BCG) [14] is a variant of the Fully-Corrective Frank-Wolfe algorithm (Algorithm 2.2) where the reoptimization step in Line 5 is solved incompletely until an accuracy ϕ_t is reached. The quantity ϕ_t is adaptively updated based on the Frank-Wolfe duality gap. The reoptimization steps require storage of the decomposition of the iterates but they can be solved efficiently using a gradient method

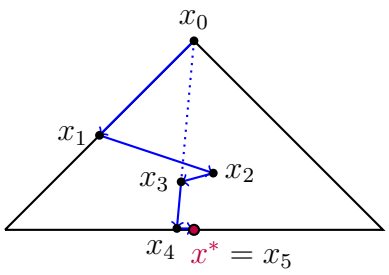

Figure 4.2: AFW breaks the zig-zagging trajectory by performing away steps. Here, x_4 is obtained using an away step which enables $x_5 = x^*$ and speeds up the algorithm. The zig-zagging is due to the weight of x_0 in the convex decomposition of x_1, x_2, x_3. It is freed up when computing x_4.

Algorithm 4.2 Decomposition-Invariant Pairwise Conditional Gradient (DICG)

Input: Start point $x_0 \in \mathcal{C}$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad v_t \leftarrow \arg\min_{v\in\mathcal{C}} \langle v, \tilde{\nabla} f(x_t)\rangle$

3: $\quad [\tilde{\nabla} f(x_t)]_i \leftarrow \begin{cases} [\nabla f(x_t)]_i & \textbf{if } [x_t]_i > 0 \\ -\infty & \textbf{if } [x_t]_i = 0 \end{cases}$ **for** $i = 1$ **to** n

4: $\quad a_t \leftarrow \arg\max_{v\in\mathcal{C}} \langle v, \tilde{\nabla} f(x_t)\rangle$

5: $\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - a_t)$

6: **end for**

[14, Sec. 4]. By blending FW steps with approximate FCFW steps, BCG generates iterates that have higher sparsity than FW or AFW and that converge at a rate $\mathcal{O}(\exp(-(S/C^{\Delta})t))$, where C^{Δ} is the *simplicial curvature* of f over the polytope.

4.3 Boosting Frank-Wolfe

4.3.1 Intuition

In the simple example of Figure 4.1, the gradient of the objective is $\nabla f(x_t) = x_t - x^*$ so $-\nabla f(x_t)$ actually points exactly towards the solution. However, FW moves in directions that are more and more orthogonal to it. Generally speaking, a rule-of-thumb in first-order optimization is to follow the direction of steepest descent, i.e., the direction of the negative gradient $-\nabla f(x_t)$. Suppose that $-\nabla f(x_t) \in \operatorname{cone}(\mathcal{C} - x_t)$ and that we are able to compute its conical decomposition, i.e., we have $-\nabla f(x_t) = \sum_{k=0}^{K_t-1} \lambda_k(v_k - x_t)$ where

$\lambda_0, \ldots, \lambda_{K_t-1} > 0$ and $v_0, \ldots, v_{K_t-1} \in \mathcal{C}$. Then by normalizing by $\Lambda_t = \sum_{k=0}^{K_t-1} \lambda_k$, we obtain a *feasible* descent direction $g_t = (1/\Lambda_t) \sum_{k=0}^{K_t-1} \lambda_k (v_k - x_t)$ in the sense that $[x_t, x_t + g_t] \subset \mathcal{C}$. Therefore, building x_{t+1} as a convex combination of x_t and $x_t + g_t$ ensures that $x_{t+1} \in \mathcal{C}$ and the projection-free property holds as in a typical FW step, all the while moving in the direction of the negative gradient $-\nabla f(x_t)$.

In practice however, computing the exact conical decomposition of $-\nabla f(x_t)$, even when this is feasible, is not necessary and it may be overkill. Indeed, all we want is to build a vector g_t such that:

(i) g_t is better aligned with $-\nabla f(x_t)$ than is the FW descent direction $v_t - x_t$, i.e.,

$$\frac{\langle g_t, -\nabla f(x_t) \rangle}{\|g_t\|_2 \|\nabla f(x_t)\|_2} \geqslant \frac{\langle v_t - x_t, -\nabla f(x_t) \rangle}{\|v_t - x_t\|_2 \|\nabla f(x_t)\|_2};$$

(ii) g_t allows a projection-free update of x_t, i.e.,

$$x_t + \gamma g_t \in \mathcal{C} \quad \text{for all } \gamma \in [0, 1],$$

or, equivalently by convexity, $[x_t, x_t + g_t] \subset \mathcal{C}$.

Thus, we propose to chase the direction of $-\nabla f(x_t)$ by sequentially picking up vertices in a matching pursuit-style [96]. This procedure is illustrated in Figure 4.3. At each round k, the procedure looks to reduce the residual r_k by subtracting its projection $\lambda_k u_k$ onto the principal component u_k. A scaling in the last round ensures that the feasibility property (ii) is satisfied.

Remark 4.1. *The procedure implicitly addresses the cone constrained quadratic optimization subproblem*

$$\min_{d \in \text{cone}(\mathcal{C} - x_t)} \frac{1}{2} \|d - (-\nabla f(x_t))\|_2^2 \tag{4.2}$$

via the Non-Negative Matching Pursuit algorithm [90], without however the aim of solving it. In particular, we would like to emphasize that it is no concern if $\|d - (-\nabla f(x_t))\|_2$ *is arbitrarily large.*

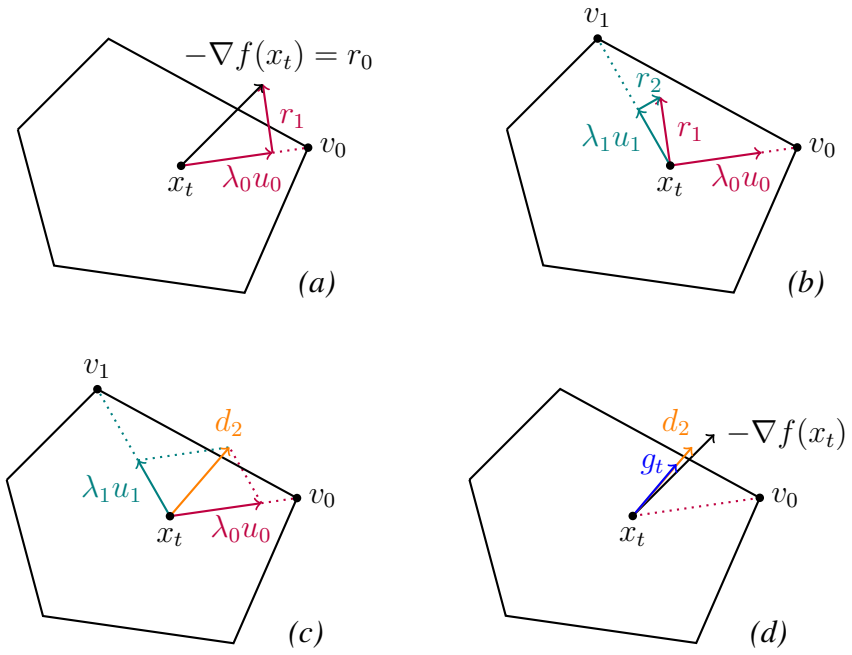

Figure 4.3: Illustration of the boosting procedure. It builds a descent direction g_t better aligned with the negative gradient direction $-\nabla f(x_t)$, while the FW descent direction is that of $v_0 - x_t$. *(a)*: Defining $r_0 \leftarrow -\nabla f(x_t)$, set $v_0 \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, -r_0 \rangle$, $u_0 \leftarrow v_0 - x_t$, and $\lambda_0 \leftarrow (\langle v_0 - x_t, r_0 \rangle / \|v_0 - x_t\|_2^2)(v_0 - x_t)$. The new residual is $r_1 \leftarrow r_0 - \lambda_0 u_0$. *(b)*: Repeat *(a)* with r_1. *(c)*: Set $d_2 \leftarrow \lambda_0 u_0 + \lambda_1 u_1$. *(d)*: Scale d_2 to obtain $g_t \leftarrow d_2/(\lambda_0 + \lambda_1)$. Note that $[x_t, x_t + d_2] \not\subset \mathcal{C}$ but $[x_t, x_t + g_t] \subset \mathcal{C}$. Moving along the segment $[x_t, x_t + g_t]$ ensures that $x_{t+1} \in \mathcal{C}$.

4.3.2 The Boosted Frank-Wolfe algorithm

The Boosted Frank-Wolfe algorithm (BoostFW) is presented in Algorithm 4.3, where the boosting procedure takes place in Lines 2–20. In short, it replaces the linear minimization oracle in FW (Algorithm 2.1). The comparison $\langle v_k - x_t, r_k \rangle$ vs. $\langle -d_k/\|d_k\|, r_k \rangle$ in Line 8 is less intuitive than the rest of the procedure, illustrated in Figure 4.3, but it is necessary to ensure convergence; see [90].

Since we are only interested in the direction of $-\nabla f(x_t)$, the stopping criterion in the

procedure (Line 11) is an alignment condition between $-\nabla f(x_t)$ and the current estimated direction d_k, which serves as descent direction for BoostFW. We measure the alignment between a target direction $d \in \mathbb{R}^n \backslash \{0\}$ and its estimate $\hat{d} \in \mathbb{R}^n$ via the cosine similarity, defined in (4.3). It is invariant by scaling of d or $\hat{d}$, and the higher the value, the better the alignment:

$$\cos(\hat{d}, d) = \begin{cases} \dfrac{\langle \hat{d}, d \rangle}{\|\hat{d}\|_2 \|d\|_2} & \text{if } \hat{d} \neq 0, \\ -1 & \text{if } \hat{d} = 0. \end{cases} \tag{4.3}$$

In order to optimize the trade-off between progress and complexity per iteration, we allow for (very) inexact alignments and we stop the procedure as soon as *sufficient* progress in alignment is not met (Lines 14–16). Furthermore, note that it is not possible to obtain a perfect alignment when $-\nabla f(x_t) \notin \operatorname{cone}(\mathcal{C} - x_t)$, but this is not an issue as we only seek to better align the descent direction. The number of pursuit rounds at iteration t is denoted by K_t (Line 19). In the experiments (Section 4.4), we typically set $\delta \leftarrow 10^{-3}$ and $K \leftarrow +\infty$; the role of K is only to cap the number of pursuit rounds per iteration when the FW oracle is particularly expensive (see Section 4.4.4). Note that if $K = 1$, then BoostFW reduces to FW.

We present in Proposition 4.2 some properties satisfied by BoostFW.

Proposition 4.2. *Let $t \in \mathbb{N}$ and suppose that $x_t \in \mathcal{C}$. Then:*

(i) d_1 *is defined and* $K_t \geqslant 1$,

(ii) $\lambda_0, \ldots, \lambda_{K_t - 1} \geqslant 0$,

(iii) $d_k \in \operatorname{cone}(\mathcal{C} - x_t)$ *for all* $k \in [\![0, K_t]\!]$,

(iv) $x_t + g_t \in \mathcal{C}$ *and* $x_{t+1} \in \mathcal{C}$,

(v) $\cos(g_t, -\nabla f(x_t)) \geqslant \cos(v_t - x_t, -\nabla f(x_t)) + (K_t - 1)\delta$ *where* $v_t \in \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$ *and* $\cos(v_t - x_t, -\nabla f(x_t)) \geqslant 0$.

Algorithm 4.3 Boosted Frank-Wolfe (BoostFW)

Input: Start point $x_0 \in \mathcal{C}$, maximum number of rounds $K \in \mathbb{N}\backslash\{0\}$, alignment improvement tolerance $\delta \in]0, 1[$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad d_0 \leftarrow 0$
3: $\quad \Lambda_t \leftarrow 0$
4: $\quad$ flag $\leftarrow$ **false**
5: $\quad$ **for** $k = 0$ **to** $K - 1$ **do**
6: $\quad\quad r_k \leftarrow -\nabla f(x_t) - d_k$ ▷ kth residual
7: $\quad\quad v_k \leftarrow \arg\max_{v\in\mathcal{C}} \langle v, r_k\rangle$ ▷ FW oracle
8: $\quad\quad u_k \leftarrow \arg\max_{u\in\{v_k - x_t, -d_k/\|d_k\|_2\}} \langle u, r_k\rangle$
9: $\quad\quad \lambda_k \leftarrow \dfrac{\langle u_k, r_k\rangle}{\|u_k\|_2^2}$
10: $\quad\quad d'_k \leftarrow d_k + \lambda_k u_k$
11: $\quad\quad$ **if** $\cos(d'_k, -\nabla f(x_t)) - \cos(d_k, -\nabla f(x_t)) \geqslant \delta$ **then**
12: $\quad\quad\quad d_{k+1} \leftarrow d'_k$
13: $\quad\quad\quad \Lambda_t \leftarrow \begin{cases} \Lambda_t + \lambda_k & \textbf{if } u_k = v_k - x_t \\ \Lambda_t(1 - \lambda_k/\|d_k\|_2) & \textbf{if } u_k = -d_k/\|d_k\|_2 \end{cases}$
14: $\quad\quad$ **else**
15: $\quad\quad\quad$ flag $\leftarrow$ **true**
16: $\quad\quad\quad$ **break** ▷ exit k-loop
17: $\quad\quad$ **end if**
18: $\quad$ **end for**
19: $\quad K_t \leftarrow k$ **if** flag $=$ **true else** K
20: $\quad g_t \leftarrow d_{K_t}/\Lambda_t$ ▷ normalization
21: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$
22: **end for**

Proof. (i) We have $d_0 = 0$ so $r_0 = -\nabla f(x_t)$ and $\cos(d_0, r_0) = -1$ by definition (4.3). Furthermore, since $v_0 \in \arg\max_{v\in\mathcal{V}}\langle v, r_0\rangle$ and $x_t \in \mathcal{C}$, we have

$$\langle v_0 - x_t, r_0\rangle = \langle v_0, r_0\rangle - \langle x_t, r_0\rangle \geqslant 0,$$

so $u_0 = v_0 - x_t$ by Line 8 since $d_0 = 0$. Thus, $\lambda_0 \geqslant 0$ (Line 9) and $d'_1 = \lambda_0(v_0 - x_t)$

(Line 10) so

$$\begin{aligned}\cos(d_1', -\nabla f(x_t)) &= \frac{\langle d_1', -\nabla f(x_t)\rangle}{\|d_1'\|_2\|\nabla f(x_t)\|_2} \\ &= \frac{\langle v_0 - x_t, r_0\rangle}{\|v_0 - x_t\|_2\|r_0\|_2} \\ &\geqslant 0 \\ &\geqslant -1 + \delta \\ &= \cos(d_0, -\nabla f(x_t)) + \delta.\end{aligned}$$

Therefore, by Line 11 the gradient pursuit procedure continues.

(ii) Let $k \in [\![0, K_t - 1]\!]$. Since $v_k \in \arg\max_{v\in\mathcal{C}}\langle v, r_k\rangle$ (Line 7) and $x_t \in \mathcal{C}$,

$$\langle v_k - x_t, r_k\rangle = \max_{v\in\mathcal{C}} \langle v, r_k\rangle - \langle x_t, r_k\rangle \geqslant 0.$$

Thus, by Lines 8-9 we have

$$\lambda_k = \frac{\langle u_k, r_k\rangle}{\|u_k\|_2^2} \geqslant 0.$$

Furthermore, note that $-r_k$ is the gradient of the objective function in subproblem (4.2) and $\langle u_k, r_k\rangle$ is a scaled upper bound on its primal gap [90]. Thus, if $\lambda_k = 0$ then the gradient pursuit procedure has already converged.

(iii) We show by induction that $d_k \in \operatorname{cone}(\mathcal{C} - x_t)$ for all $k \in [\![0, K_t]\!]$. We have $d_0 = 0 \in \operatorname{cone}(\mathcal{C} - x_t)$ so the base case is satisfied. Suppose that $d_k \in \operatorname{cone}(\mathcal{C} - x_t)$ for some $k \in [\![0, K_t - 1]\!]$. If $u_k = v_k - x_t$ then $u_k \in \mathcal{C} - x_t$ and since $\lambda_k \geqslant 0$ by (ii), we have $d_{k+1} = d_k + \lambda_k(v_k - x_t) \in \operatorname{cone}(\mathcal{C} - x_t)$. Else, $u_k = -d_k/\|d_k\|_2$ so $d_{k+1} = (1 - \lambda_k/\|d_k\|_2)d_k$ and it remains to show that $1 - \lambda_k/\|d_k\|_2 \geqslant 0$. We will

show that $1 - \lambda_k / \|d_k\|_2 \geqslant 1/2$. We have

$$1 - \frac{\lambda_k}{\|d_k\|_2} \geqslant \frac{1}{2} \Leftrightarrow \frac{1}{2} \geqslant \frac{\lambda_k}{\|d_k\|_2} = \frac{\langle -d_k/\|d_k\|_2, r_k \rangle}{\|d_k\|_2}$$
$$\Leftrightarrow \frac{\|d_k\|_2^2}{2} \geqslant \langle -d_k, r_k \rangle, \tag{4.4}$$

so it suffices to show that $\|d_k\|_2^2/2 \geqslant \langle -d_k, r_k \rangle$. Now, the procedure satisfies for all $k' \in [\![0, K_t - 1]\!]$,

$$\begin{aligned}
\|r_{k'+1}\|_2^2 &= \|r_{k'} - \lambda_{k'} u_{k'}\|_2^2 \\
&= \|r_{k'}\|_2^2 - 2\lambda_{k'} \langle u_{k'}, r_{k'} \rangle + \lambda_{k'}^2 \|u_{k'}\|_2^2 \\
&= \|r_{k'}\|_2^2 - \frac{\langle u_{k'}, r_{k'} \rangle^2}{\|u_{k'}\|_2^2} \\
&\leqslant \|r_{k'}\|_2^2,
\end{aligned}$$

where we used $\lambda_{k'} = \langle u_{k'}, r_{k'} \rangle / \|u_{k'}\|_2^2$. Thus $\|r_k\|_2^2 \leqslant \|r_0\|_2^2$, i.e., since $d_0 = 0$, $\|\nabla f(x_t) + d_k\|_2^2 \leqslant \|\nabla f(x_t)\|_2^2$ so

$$\|\nabla f(x_t)\|_2^2 \geqslant \|\nabla f(x_t) + d_k\|_2^2 = \|\nabla f(x_t)\|_2^2 + 2\langle d_k, \nabla f(x_t) \rangle + \|d_k\|_2^2,$$

hence

$$\langle d_k, \nabla f(x_t) \rangle \leqslant -\frac{\|d_k\|_2^2}{2}.$$

Thus,

$$\begin{aligned}
\langle -d_k, r_k \rangle &= \langle d_k, \nabla f(x_t) + d_k \rangle \\
&= \langle d_k, \nabla f(x_t) \rangle + \|d_k\|_2^2 \\
&\leqslant \frac{\|d_k\|_2^2}{2}.
\end{aligned}$$

Therefore, with (4.4) we can conclude that $d_{k+1} \in \text{cone}(\mathcal{C} - x_t)$.

(iv) By (iii), $d_{K_t} \in \text{cone}(\mathcal{C} - x_t)$ so since $g_t = d_{K_t}/\Lambda_t$, to show that $x_t + g_t \in \mathcal{C}$ it suffices to show that the sum of coefficients in the conical decomposition of d_{K_t} is equal to Λ_t, and then it follows that $g_t \in \text{conv}(\mathcal{C} - x_t) = \text{conv}(\mathcal{C}) - x_t = \mathcal{C} - x_t$. By Line 13, this is true and is verified by a simple induction on k: the base case is satisfied and if $u_k = v_k - x_t$ then $d_{k+1} = d_k + \lambda_k(v_k - x_t)$ and Line 13 shows that $\Lambda_t \leftarrow \Lambda_t + \lambda_k$ is updated accordingly, else $u_k = -d_k/\|d_k\|_2$ so $d_{k+1} = (1 - \lambda_k/\|d_k\|_2)d_k$ and Line 13 shows that $\Lambda_t \leftarrow \Lambda_t(1 - \lambda_k/\|d_k\|_2)$ is again updated accordingly. Thus, $x_t + g_t \in \mathcal{C}$. Then,

$$\begin{aligned} x_{t+1} &= x_t + \gamma_t g_t \\ &= x_t + \gamma_t((x_t + g_t) - x_t) \\ &= (1 - \gamma_t)x_t + \gamma_t \underbrace{(x_t + g_t)}_{\in \mathcal{C}}. \end{aligned}$$

Since $\gamma_t \in [0, 1]$, we conclude that $x_{t+1} \in \mathcal{C}$ by convex combination.

(v) Since $d_0 = 0$, we have $r_0 = -\nabla f(x_t)$ so by Line 7, $v_0 \in \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(x_t) \rangle$, and we have $d_1 = v_t - x_t$. Let $v_t v_0$. Since $g_t = d_{K_t}/\Lambda_t$ where $K_t \geqslant 1$ by (i), by Line 11 we obtain

$$\begin{aligned} \cos(g_t, -\nabla f(x_t)) &= \cos(d_{K_t}, -\nabla f(x_t)) \\ &\geqslant \cos(v_t - x_t, -\nabla f(x_t)) + (K_t - 1)\delta. \end{aligned}$$

Lastly,

$$\begin{aligned} \cos(v_t - x_t, -\nabla f(x_t)) &= \frac{\langle x_t - v_t, \nabla f(x_t) \rangle}{\|x_t - v_t\|_2 \|\nabla f(x_t)\|_2} \\ &\geqslant 0, \end{aligned}$$

because $\langle x_t - v_t, \nabla f(x_t) \rangle \geqslant 0$ since $x_t \in \mathcal{C}$.

□

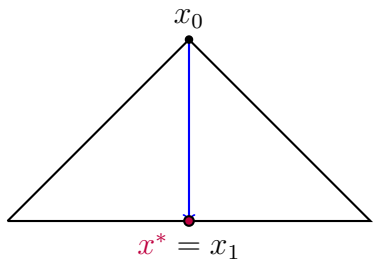

Figure 4.4: We complete the series of Figures 4.1–4.2. Here, BoostFW can exactly estimate the direction of $-\nabla f(x_0) = -(x_0 - x^*)$ in just two rounds, and it converges in 1 iteration.

4.3.3 Convergence analysis

Let $(x_t)_{t \in \mathbb{N}}$ denote the sequence of iterates generated by BoostFW. We consider the initialization

$$x_0 \in \arg\min_{v \in \mathcal{C}} \langle v, \nabla f(y) \rangle \quad \text{for a given } y \in \mathcal{C}, \tag{4.5}$$

which guarantees, similarly to Lemma 2.5, that

$$f(x_0) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2}. \tag{4.6}$$

The step-size strategy is similar to the closed-loop strategy (2.2) used in FW, where $v_t - x_t$ is replaced with the boosted direction g_t better aligned with $-\nabla f(x_t)$:

$$\gamma_t \leftarrow \min \left\{ \frac{\langle g_t, -\nabla f(x_t) \rangle}{L \|g_t\|_2^2}, 1 \right\}. \tag{4.7}$$

Theorem 4.3 provides a quantitative estimation of the convergence of BoostFW.

Theorem 4.3. *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth, convex, μ-gradient dominated function.*

Consider the initialization (4.5) *and the closed-loop strategy* (4.7)*. Then for all* $t \in \mathbb{N}$*,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2} \prod_{s=0}^{t-1} \left(1 - \eta_s^2 \frac{\mu}{L}\right)^{1_{\{\gamma_s<1\}}} \left(1 - \frac{\|g_s\|_2}{2\|v_s - x_s\|_2}\right)^{1_{\{\gamma_s=1\}}}$$

where $\eta_s = \cos(g_s, -\nabla f(x_s))$ *and* $v_s \in \arg\min_{v\in\mathcal{C}} \langle v, \nabla f(x_s)\rangle$ *for all* $s \in \mathbb{N}$*.*

Proof. Let $\varepsilon_t = f(x_t) - \min_{\mathcal{C}} f$ for all $t \in \mathbb{N}$. By (4.6),

$$\varepsilon_0 \leqslant \frac{LD^2}{2},$$

so the base case is satisfied. Let $t \in \mathbb{N}$. We have

$$\begin{aligned} \eta_t &= \cos(g_t, -\nabla f(x_t)) \\ &= \frac{\langle g_t, -\nabla f(x_t)\rangle}{\|g_t\|_2 \|\nabla f(x_t)\|_2}. \end{aligned}$$

Suppose that $\gamma_t = \langle g_t, -\nabla f(x_t)\rangle / (L\|g_t\|_2^2)$. Then since f is L-smooth and is μ-gradient dominated,

$$\begin{aligned} \varepsilon_{t+1} &\leqslant \varepsilon_t + \gamma_t \langle g_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma_t^2 \|g_t\|_2^2 \\ &= \varepsilon_t - \frac{\langle g_t, \nabla f(x_t)\rangle^2}{2L\|g_t\|_2^2} \\ &= \varepsilon_t - \eta_t^2 \frac{\|\nabla f(x_t)\|_2^2}{2L} \\ &\leqslant \varepsilon_t - \eta_t^2 \frac{2\mu\varepsilon_t}{2L} \\ &= \left(1 - \eta_t^2 \frac{\mu}{L}\right) \varepsilon_t. \end{aligned} \tag{4.8}$$

Else, $\langle g_t, -\nabla f(x_t)\rangle / (L\|g_t\|_2^2) > 1$ and $\gamma_t = 1$. Thus,

$$L\|g_t\|^2 < \langle g_t, -\nabla f(x_t)\rangle,$$

so

$$\begin{aligned}\varepsilon_{t+1} &\leqslant \varepsilon_t + \gamma_t\langle g_t, \nabla f(x_t)\rangle + \frac{L}{2}\gamma_t^2\|g_t\|^2 \\ &= \varepsilon_t + \langle g_t, \nabla f(x_t)\rangle + \frac{L}{2}\|g_t\|_2^2 \\ &< \varepsilon_t + \frac{\langle g_t, \nabla f(x_t)\rangle}{2}. \end{aligned} \tag{4.9}$$

Recall that $g_t = d_{K_t}/\Lambda_t$ and $d_1 = \lambda_0(v_0 - x_t)$ where $v_0 \in \arg\min_{v\in\mathcal{C}}\langle v, \nabla f(x_t)\rangle$. If $K_t = 1$ then $d_{K_t} = d_1$, else $\cos(d_{K_t}, -\nabla f(x_t)) > \cos(d_1, -\nabla f(x_t))$ by the condition in Line 11. In both cases, we obtain

$$\frac{\langle g_t, -\nabla f(x_t)\rangle}{\|g_t\|_2} \geqslant \frac{\langle v_0 - x_t, -\nabla f(x_t)\rangle}{\|v_0 - x_t\|_2}. \tag{4.10}$$

Let $x^* \in \arg\min_{\mathcal{C}} f$. By convexity of f and optimality of v_0,

$$\begin{aligned}\varepsilon_t &= f(x_t) - f(x^*) \\ &\leqslant \langle x_t - x^*, \nabla f(x_t)\rangle \\ &\leqslant \langle x_t - v_0, \nabla f(x_t)\rangle. \end{aligned} \tag{4.11}$$

Thus, with (4.9) and (4.10), we have

$$\varepsilon_{t+1} < \left(1 - \frac{\|g_t\|_2}{2\|v_0 - x_t\|_2}\right)\varepsilon_t.$$

Therefore, together with (4.8) we conclude that for all $t \in \mathbb{N}$,

$$\varepsilon_t \leqslant \varepsilon_0 \prod_{s=0}^{t-1}\left(1 - \eta_s^2\frac{\mu}{L}\right)^{1_{\{\gamma_s<1\}}}\left(1 - \frac{\|g_s\|_2}{2\|v_s - x_s\|_2}\right)^{1_{\{\gamma_s=1\}}},$$

where we denote $v_s \in \arg\min_{v\in\mathcal{C}}\langle v, \nabla f(x_s)\rangle$ for all $s \in \mathbb{N}$. □

Note that $\gamma_t = 1$ is extremely rare in practice, and we observed no more than 1 such

iteration in each of the experiments (Section 4.4). This is a similar phenomenon to that in the Away-Step and Pairwise Frank-Wolfe algorithms [78]. Similarly, $K_t > 1$ simply means that it is possible to increase the alignment by δ twice and consecutively, where δ is typically set to a low value. In the experiments, we set $\delta \leftarrow 10^{-3}$ and we observed $K_t > 1$ (or even $K_t > 5$) almost everytime. Thus, $N_t \approx t$ where $N_t = \operatorname{card}\{s \in [\![0, t-1]\!] \mid \gamma_s < 1, K_s > 1\}$ denotes the number of "good iterations" until iteration t. We introduce the quantity N_t in order to write a *fully explicit* convergence rate of BoostFW. In Theorem 4.4, we use a weaker assumption $N_t = \Omega(t)$.

Theorem 4.4. *Let $f\colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth, convex, μ-gradient dominated function. Consider the initialization* (4.5) *and the closed-loop strategy* (4.7)*, and suppose that $N_t \geqslant \omega t$ for all $t \in \mathbb{N}$, where $\omega > 0$. Then for all $t \in \mathbb{N}$,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2} \exp\left(-\delta^2 \frac{\mu}{L} \omega t\right).$$

Proof. Let $t \in \mathbb{N}$. We have $N_t \geqslant \omega t$ and, by Proposition 4.2(v), if $K_t > 1$ then $\eta_t \geqslant \delta$. By Theorem 4.3,

$$\begin{aligned}
f(x_t) - \min_{\mathcal{C}} f &\leqslant \frac{LD^2}{2} \prod_{s=0}^{t-1} \left(1 - \eta_s^2 \frac{\mu}{L}\right)^{1_{\{\gamma_s < 1\}}} \left(1 - \frac{\|g_s\|_2}{2\|v_s - x_s\|_2}\right)^{1_{\{\gamma_s = 1\}}} \\
&\leqslant \frac{LD^2}{2} \prod_{\substack{s=0 \\ \gamma_s < 1 \\ K_s > 1}}^{t-1} \left(1 - \eta_s^2 \frac{\mu}{L}\right) \\
&\leqslant \frac{LD^2}{2} \left(1 - \delta^2 \frac{\mu}{L}\right)^{N_t} \\
&\leqslant \frac{LD^2}{2} \exp\left(-\delta^2 \frac{\mu}{L} N_t\right) \\
&\leqslant \frac{LD^2}{2} \exp\left(-\delta^2 \frac{\mu}{L} \omega t\right).
\end{aligned}$$

□

Remark 4.5. *Note that if $K_t = 1$ at every iteration than BoostFW reduces to FW, and the*

convergence rate is $\mathcal{O}(1/t)$ (Theorem 2.6).

4.3.4 Extension to Frank-Wolfe variants

Before moving on to the computational experiments, we briefly show here that the boosting procedure can be written in a very generic form (Algorithm 4.4), where the target direction and the reference point are denoted by $\mathbf{d} \in \mathbb{R}^n \backslash \{0\}$ and $\mathbf{z} \in \mathcal{C}$ respectively. In the case of FW, we have $\mathbf{d} \leftarrow -\nabla f(x_t)$ and $\mathbf{z} \leftarrow x_t$.

Algorithm 4.4 Boosting procedure Boost$(\mathbf{d}, \mathbf{z}, K, \delta)$

Input: Target direction $\mathbf{d} \in \mathbb{R}^n \backslash \{0\}$, reference point $\mathbf{z} \in \mathcal{C}$, maximum number of rounds $K \in \mathbb{N} \backslash \{0\}$, alignment improvement tolerance $\delta \in]0, 1[$.
Output: Boosted descent direction $g \in \mathcal{C} - z$.

$d_0 \leftarrow 0$
$\Lambda \leftarrow 0$
flag $\leftarrow$ **false**
for $k = 0$ **to** $K - 1$ **do**
  $r_k \leftarrow \mathbf{d} - d_k$ ▷ kth residual
  $v_k \leftarrow \arg\max_{v \in \mathcal{C}} \langle v, r_k \rangle$ ▷ FW oracle
  $u_k \leftarrow \arg\max_{u \in \{v_k - \mathbf{z}, -d_k/\|d_k\|_2\}} \langle u, r_k \rangle$
  $\lambda_k \leftarrow \dfrac{\langle u_k, r_k \rangle}{\|u_k\|_2^2}$
  $d'_k \leftarrow d_k + \lambda_k u_k$
  if $\cos(d'_k, \mathbf{d}) - \cos(d_k, \mathbf{d}) \geqslant \delta$ **then**
    $d_{k+1} \leftarrow d'_k$
    $\Lambda \leftarrow \begin{cases} \Lambda + \lambda_k & \textbf{if } u_k = v_k - \mathbf{z} \\ \Lambda(1 - \lambda_k / \|d_k\|_2) & \textbf{if } u_k = -d_k / \|d_k\|_2 \end{cases}$
  else
    flag $\leftarrow$ **true**
    break ▷ exit k-loop
  end if
end for
$K' \leftarrow k$ **if** flag $=$ **true else** K
$g \leftarrow d_{K'} / \Lambda$ ▷ normalization

BoostFW can thus be written as in Algorithm 4.5.

Algorithm 4.5 Boosted Frank-Wolfe (BoostFW)

Input: Start point $x_0 \in \mathcal{C}$, maximum number of rounds $K \in \mathbb{N}\backslash\{0\}$, alignment improvement tolerance $\delta \in]0, 1[$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), x_t, K, \delta)$

3: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$

4: **end for**

The procedure can also be applied to boost other Frank-Wolfe algorithms. For example, we can apply it to boost the performance of DICG (Algorithm 4.6). Note that the reference point here is the away vertex a_t.

Algorithm 4.6 Boosted Decomposition-Invariant Pairwise Conditional Gradient (BoostDICG)

Input: Start point $x_0 \in \mathcal{C}$, maximum number of rounds $K \in \mathbb{N}\backslash\{0\}$, alignment improvement tolerance $\delta \in]0, 1[$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**

2: $\quad [\tilde{\nabla} f(x_t)]_i \leftarrow \begin{cases} [\nabla f(x_t)]_i & \textbf{if } [x_t]_i > 0 \\ -\infty & \textbf{if } [x_t]_i = 0 \end{cases} \quad \textbf{for } i = 1 \textbf{ to } n$

3: $\quad a_t \leftarrow \arg\max_{v\in\mathcal{C}} \langle v, \tilde{\nabla} f(x_t)\rangle$

4: $\quad g_t \leftarrow \text{Boost}(-\nabla f(x_t), a_t, K, \delta)$

5: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$

6: **end for**

An example where the target direction is not that of the negative gradient $-\nabla f(x_t)$ is, e.g., when boosting a momentum variant of FW (Algorithm 4.7).

Algorithm 4.7 Boosted Momentum Frank-Wolfe

Input: Start point $x_0 \in \mathcal{C}$, momentum parameter $\beta \in]0, 1[$, maximum number of rounds $K \in \mathbb{N}\backslash\{0\}$, alignment improvement tolerance $\delta \in]0, 1[$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0, 1]$.

1: $m_{-1} \leftarrow 0$

2: **for** $t = 0$ **to** $T - 1$ **do**

3: $\quad m_t \leftarrow \beta m_{t-1} + (1 - \beta)\nabla f(x_t)$

4: $\quad g_t \leftarrow \text{Boost}(-m_t, x_t, K, \delta)$

5: $\quad x_{t+1} \leftarrow x_t + \gamma_t g_t$

6: **end for**

4.4 Computational experiments

In this section, we demonstrate that BoostFW is very efficient computationally even though it may call the linear minimization oracle multiple times during the boosting procedure. We compare BoostFW to AFW, DICG, and BCG in a series of computational experiments. We ran two strategies for AFW, one with the default line search (AFW-ls) and one using a closed-loop strategy (AFW-L):

$$\gamma_t \leftarrow \begin{cases} \min\left\{\dfrac{\langle x_t - v_t, \nabla f(x_t)\rangle}{L\|x_t - v_t\|_2^2}, 1\right\} & \text{if FW step} \\ \min\left\{\dfrac{\langle a_t - x_t, \nabla f(x_t)\rangle}{L\|a_t - x_t\|_2^2}, \gamma_{\max}\right\} & \text{if away step,} \end{cases}$$

where $\gamma_{\max}$ is defined in AFW (Algorithm 4.1). Both strategies yield the same convergence rate [111]. For BoostFW, we also ran a line search strategy to demonstrate that the speedup really comes from the boosting procedure and not from being line search-free. Results further show that the closed-loop strategy (4.7) is very performant in CPU time. The line search-free strategy of DICG is not competitive in the experiments.

DICG is not applicable to optimization problems over the ℓ_1-ball

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \ & f(x) \\ \text{s.t. } & \|x\|_1 \leqslant \tau, \end{aligned} \tag{4.12}$$

however we can perform a change of variables $x_i = z_i - z_{n+i}$ and use the following reformulation over the simplex:

$$\begin{aligned} \min_{z \in \mathbb{R}^{2n}} \ & f([z]_{1:n} - [z]_{n+1:2n}) \\ \text{s.t. } & z \in \tau \Delta_{2n}, \end{aligned} \tag{4.13}$$

where $[z]_{1:n}$ and $[z]_{n+1:2n}$ denote the truncation to $\mathbb{R}^n$ of the first n entries and the last n entries of $z \in \mathbb{R}^{2n}$ respectively. Fact 4.6 formally states the equivalence between problems (4.12) and (4.13).

Fact 4.6. *Consider $\mathbb{R}^n$ and let $\tau > 0$. Then $\mathcal{B}_1(0, \tau) = \{[z]_{1:n} - [z]_{n+1:2n} \mid z \in \tau \Delta_{2n}\}$.*

Proof. Let $x \in \mathcal{B}_1(0, \tau)$. Define $\delta = (\tau - \|x\|_1)/2n \geqslant 0$ and $z \in \mathbb{R}^{2n}$ by

$$[z]_i = \begin{cases} [x]_i + \delta & \text{if } [x]_i \geqslant 0 \\ \delta & \text{if } [x]_i < 0 \end{cases} \quad \text{and} \quad [z]_{n+i} = \begin{cases} \delta & \text{if } [x]_i \geqslant 0 \\ -[x]_i + \delta & \text{if } [x]_i < 0 \end{cases}$$

for all $i \in [\![1, n]\!]$. Then $x = [z]_{1:n} - [z]_{n+1:2n}$ and $z \geqslant 0$. Furthermore,

$$\begin{aligned} 1^\top z &= \sum_{i=1}^{n} ([z]_i + [z]_{n+i}) \\ &= \sum_{i=1}^{n} (|[x]_i| + 2\delta) \\ &= \|x\|_1 + 2n\delta \\ &= \tau \end{aligned}$$

so $z \in \tau\Delta_{2n}$. For the reverse direction, let $z \in \tau\Delta_{2n}$ and $x = [z]_{1:n} - [z]_{n+1:2n}$. Then

$$\begin{aligned}\|x\|_1 &= \sum_{i=1}^{n} |[z]_i - [z]_{n+i}| \\ &\leqslant \sum_{i=1}^{n} ([z]_i + [z]_{n+i}) \\ &= \tau\end{aligned}$$

so $x \in \mathcal{B}_1(0, \tau)$. □

We implemented all the algorithms in Python using the same code framework for fair comparisons. In the case of synthetic data, we generated them from Gaussian distributions. We ran the experiments on a laptop under Linux Ubuntu 18.04 with Intel Core i7 3.5GHz CPU and 8GB RAM. The code is available at https://github.com/cyrillewcombettes/boostfw. In each experiment, we estimated the smoothness constant L of the (convex) objective function $f\colon \mathbb{R}^n \to \mathbb{R}$, i.e., the Lipschitz constant of the gradient function $\nabla f\colon \mathbb{R}^n \to \mathbb{R}^n$, by sampling a few pairs of points $(x, y) \in \mathcal{C} \times \mathcal{C}$ and computing an upper bound on $\|\nabla f(y) - \nabla f(x)\|_2 / \|y - x\|_2$. Unless specified otherwise, we set $\delta \leftarrow 10^{-3}$ and $K \leftarrow +\infty$ in BoostFW. The role of K is only to cap the number of pursuit rounds per iteration when the FW oracle is particularly expensive (see Section 4.4.4).

4.4.1 Lower bound on the number of oracle calls

In Figure 4.5, we demonstrate that although BoostFW may call the oracle multiple times per iteration, it is still compatible with the lower bound from Proposition 2.8. We set $n = 1\,000$ and since the objective is quadratic, we used an exact line search step-size strategy in FW-ls and BoostFW-ls. Note that the optimal value of the problem is $1/n = 10^{-3}$.

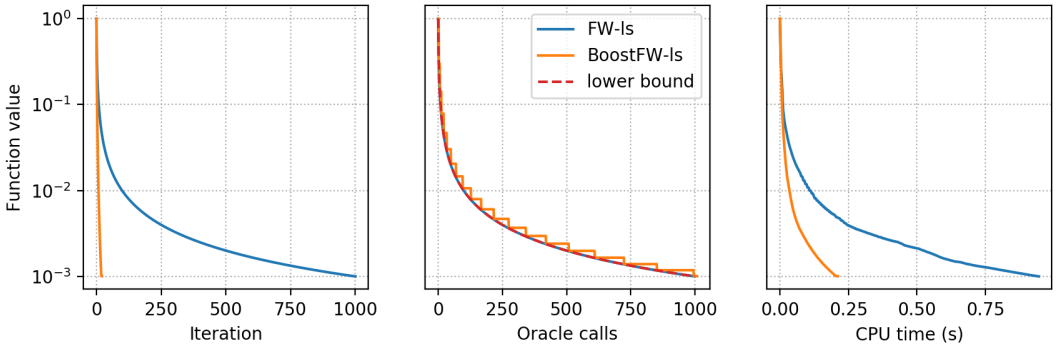

Figure 4.5: Lower bound on the number of oracle calls.

4.4.2 Sparse signal recovery

Let $x^* \in \mathbb{R}^n$ be a signal which we want to recover as a sparse representation from observations $y = Ax^* + w$, where $A \in \mathbb{R}^{m\times n}$ and $w \sim \mathcal{N}(0, \sigma^2 I_m)$ is the noise in the measurements. The natural formulation of the problem is

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \|y - Ax\|_2^2 \\ &\text{s.t. } \|x\|_0 \leqslant \|x^*\|_0 \end{aligned}$$

but the ℓ_0-pseudo-norm $\|\cdot\|_0 \colon x \in \mathbb{R}^n \mapsto \operatorname{card}\{i \in [\![1, n]\!] \mid [x]_i \neq 0\}$ is nonconvex and renders the problem intractable in many situations [103]. To remedy this, the ℓ_1-norm is often used as a convex surrogate and leads to the following lasso formulation [124] of the problem:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \|y - Ax\|_2^2 \\ &\text{s.t. } \|x\|_1 \leqslant \|x^*\|_1. \end{aligned}$$

In order to compare to DICG, which is not applicable to this formulation, we ran all algorithms on the reformulation (4.13). We set $m = 200$, $n = 500$, $\sigma = 0.05$, and $\tau = \|x^*\|_1$. Since the objective function is quadratic, we can derive a closed-form solution to the line

search and there is no need for AFW-L or BoostFW-L. The results are presented in Figure 4.6.

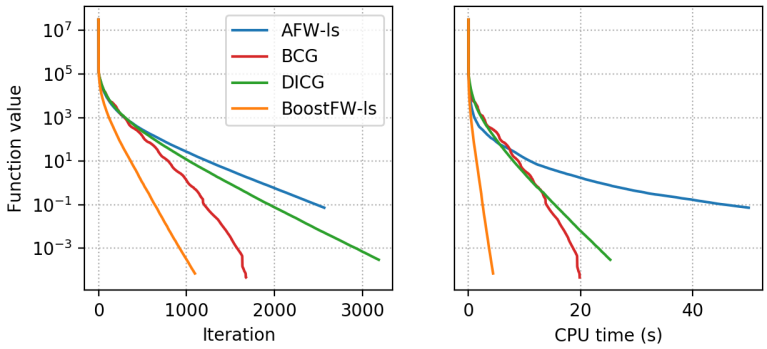

Figure 4.6: Sparse signal recovery.

4.4.3 Sparsity-constrained logistic regression

We consider the task of recognizing the handwritten digits 4 and 9 from the Gisette dataset [51], available at https://archive.ics.uci.edu/ml/datasets/Gisette. The dataset includes a high number of distractor features with no predictive power. Hence, a sparsity-constrained logistic regression model is suited for the task. The sparsity-inducing constraint is realized via the ℓ_1-norm:

$$\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \ln(1 + \exp(-y_i a_i^\top x))$$
$$\text{s.t. } \|x\|_1 \leqslant \tau$$

where $a_1, \ldots, a_m \in \mathbb{R}^n$ and $y \in \{-1, +1\}^m$. In order to compare to DICG, which is not applicable to this formulation, we ran all algorithms on the reformulation (4.13). We used $m = 2\,000$ samples and the number of features is $n = 5\,000$. We set $\tau = 10$, $L = 0.5$, and $\delta \leftarrow 10^{-4}$ in BoostFW. The results are presented in Figure 4.7. As expected, AFW-L and BoostFW-L converge faster in CPU time as they do not rely on line search, however they converge slower per iteration as each iteration provides less progress.

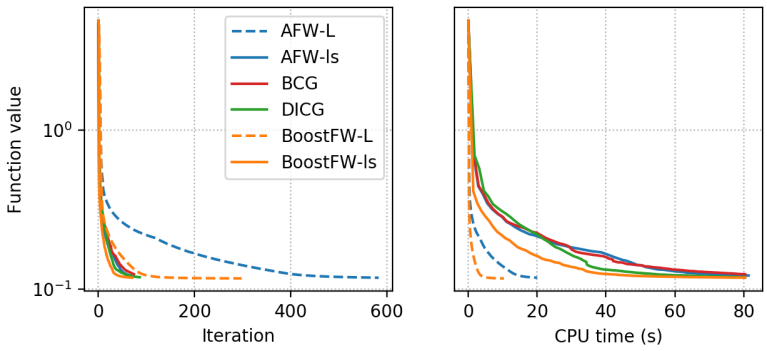

Figure 4.7: Sparse logistic regression on the Gisette dataset.

4.4.4 Traffic assignment

We consider the traffic assignment problem. The task is to assign vehicles on a traffic network in order to minimize congestion while satisfying travel demands. Let $\mathcal{A}$, $\mathcal{R}$, and $\mathcal{S}$ be the sets of links, routes, and origin-destination pairs respectively. For every pair $(i,j) \in \mathcal{S}$, let $\mathcal{R}_{i,j}$ and $d_{i,j}$ be the set of routes and the travel demand from i to j. Let x_a and t_a be the flow and the travel time on link $a \in \mathcal{A}$, and let y_r be the flow on route $r \in \mathcal{R}$. The Beckmann formulation of the problem [7], derived from the Wardrop equilibrium conditions [129], is

$$\begin{aligned} \min_{x \in \mathbb{R}^{|\mathcal{A}|}} \ & \sum_{a \in \mathcal{A}} \int_0^{x_a} t_a(\xi)\, \mathrm{d}\xi & (4.14) \\ \text{s.t. } x_a &= \sum_{r \in \mathcal{R}} 1_{\{a \in r\}} y_r \quad && a \in \mathcal{A} \\ \sum_{r \in \mathcal{R}_{i,j}} & y_r = d_{i,j} && (i,j) \in \mathcal{S} \\ y_r &\geqslant 0 && r \in \mathcal{R}_{i,j},\ (i,j) \in \mathcal{S}. \end{aligned}$$

A commonly used expression for the travel time t_a as a function of the flow x_a, developed by the Bureau of Public Records, is $t_a \colon x_a \in \mathbb{R}_+ \mapsto \tau_a(1 + 0.15(x_a/c_a)^4)$ where τ_a and c_a are the free-flow travel time and the capacity of the link. A linear minimization over the feasible region in (4.14) amounts to computing the shortest routes between all

origin-destination pairs. Thus, the FW oracle is particularly expensive here so we capped the maximum number of rounds in BoostFW to $K \leftarrow 5$ (Figure 4.9). We implemented the oracle using the function `all_pairs_dijkstra_path` from the Python package `networkx` [52]. We created a directed acyclic graph with 500 nodes split into 20 layers of 25 nodes each, and randomly dropped links with probability 0.5 so $|\mathcal{A}| \approx 6\,000$ and $\operatorname{card} \mathcal{S} \approx 113\,000$. We set $d_{i,j} \sim \mathcal{U}([0,1])$ for every $(i,j) \in \mathcal{S}$. DICG is not applicable here and AFW-L and BoostFW-L were not competitive. The results are presented in Figure 4.8.

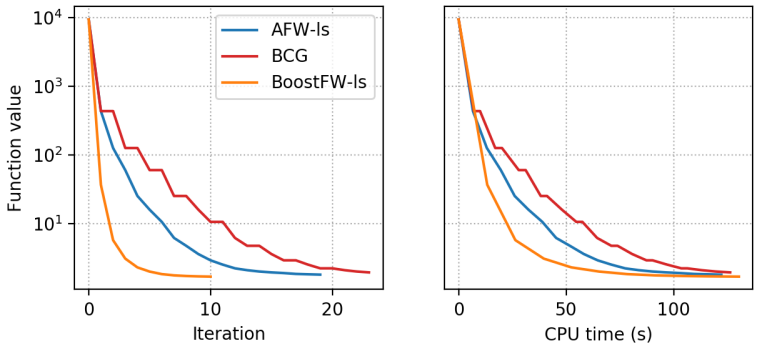

Figure 4.8: Traffic assignment.

We define the relative improvement in alignment between rounds $k-1$ and $k \in [\![2, K_t]\!]$ in the gradient pursuit procedure at iteration $t \in \mathbb{N}$ of BoostFW as

$$\theta_{t,k} = \frac{\cos(d_k, -\nabla f(x_t)) - \cos(d_{k-1}, -\nabla f(x_t))}{\cos(d_{k-1}, -\nabla f(x_t))}.$$

For a fixed round k, we plot in Figure 4.9 the mean of $\theta_{t,k}$ across all iterations t that performed a kth round, i.e.,

$$\theta_k = \frac{1}{\operatorname{card}\{t \in [\![0, T-1]\!] \mid k \leqslant K_t\}} \sum_{t=0}^{T-1} \theta_{t,k} 1_{\{k \leqslant K_t\}},$$

in the sparse signal recovery experiment (Section 4.4.2). The error bars represent ± 1 standard deviation. We see that on average the second round produces an improvement in alignment of $\sim 39\%$, the third round produces an improvement of $\sim 17\%$, etc. In particu-

lar, the plot suggests that 5 rounds in each iteration are enough.

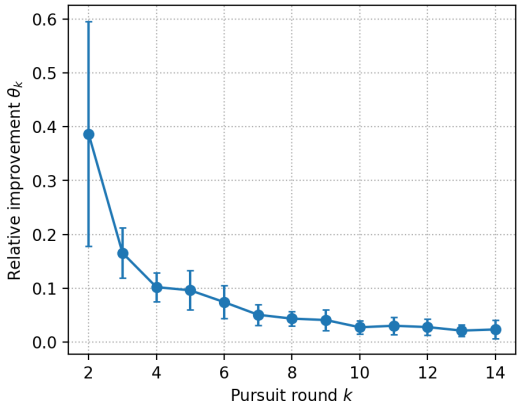

Figure 4.9: Relative improvements in alignment during the gradient pursuit procedure.

4.4.5 Collaborative filtering

We consider the task of collaborative filtering on the MovieLens 100k dataset [53], available at https://grouplens.org/datasets/movielens/100k/. The low-rank assumption on the solution and the approach of [98] lead to the following problem formulation:

$$\min_{X \in \mathbb{R}^{m \times n}} \frac{1}{|\mathcal{I}|} \sum_{(i,j) \in \mathcal{I}} h_\rho(Y_{i,j} - X_{i,j})$$
$$\text{s.t. } \|X\|_{\text{nuc}} \leqslant \tau$$

where h_ρ is the Huber loss with parameter $\rho > 0$ [61]:

$$h_\rho \colon t \in \mathbb{R} \mapsto \begin{cases} t^2/2 & \text{if } |t| \leqslant \rho \\ \rho(|t| - \rho/2) & \text{if } |t| > \rho, \end{cases}$$

$Y \in \mathbb{R}^{m \times n}$ is the given matrix to complete, $\mathcal{I} \subset [\![1, m]\!] \times [\![1, n]\!]$ is the set of indices of observed entries in Y, and $\|\cdot\|_{\text{nuc}} \colon X \in \mathbb{R}^{m \times n} \mapsto \operatorname{tr}(\sqrt{X^\top X}) = \sum_{i=1}^{\min\{m,n\}} \sigma_i(X)$ is the nuclear norm and equals the sum of the singular vectors. It serves as a convex surrogate for

the rank constraint [42]. Since

$$\{X \in \mathbb{R}^{m\times n} \mid \|X\|_{\text{nuc}} = 1\} = \text{conv}\{uv^\top \mid u \in \mathbb{R}^m, v \in \mathbb{R}^n, \|u\|_2 = \|v\|_2 = 1\},$$

a linear minimization over the nuclear norm-ball of radius τ amounts to computing the top left and right singular vectors u and v of $-\nabla f(X_t)$ and to return $\tau uv^\top$. To this end, we used the function `svds` from the Python package `scipy.sparse.linalg` [128]. We have $m = 943$, $n = 1\,682$, and $|\mathcal{I}| = 10^5$, and we set $\rho = 1$, $\tau = 5\,000$, and $L = 5 \cdot 10^{-6}$. DICG is not applicable here. The results are presented in Figure 4.10.

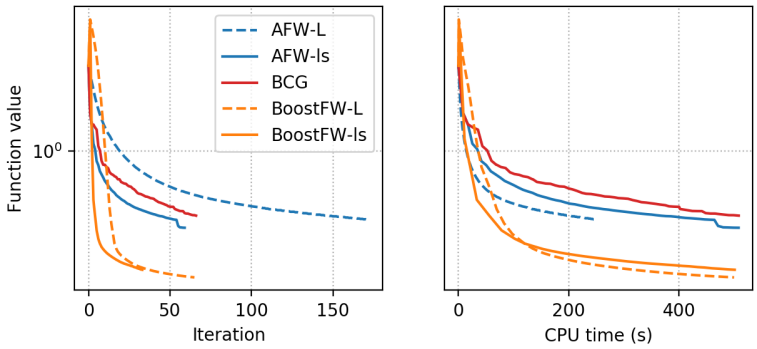

Figure 4.10: Collaborative filtering on the MovieLens 100k dataset.

The time limit here was set to 500 seconds but for AFW-L we reduced it to 250 seconds, else it raises a memory error on our machine shortly after. This is because AFW requires storing the convex decomposition of the iterate onto the vertices of $\mathcal{C}$. Note that BoostFW-ls converges faster in CPU time than AFW-L, although it relies on line search, and that BoostFW-L converges faster per iteration than the other methods although it does not rely on line search.

4.4.6 Video co-localization

We consider the task of video co-localization on the aeroplane class of the YouTube-Objects dataset [115], using the problem formulation of [66]. The goal is to localize (with bounding boxes) the aeroplane object across the video frames. It consists in minimizing $f\colon x \in$

$\mathbb{R}^{660} \mapsto x^\top Ax/2 + b^\top x$ over a flow polytope, where $A \in \mathbb{R}^{660\times 660}$, $b \in \mathbb{R}^{660}$, and the polytope each encode a part of the temporal consistency in the video frames. We obtained the data from https://github.com/Simon-Lacoste-Julien/linearFW. A linear minimization over the flow polytope amounts to computing a shortest path in the corresponding directed acyclic graph. DICG performs particularly well on this instance so we also compared to BoostDICG (Algorithm 4.6). Since the objective function is quadratic, we can derive a closed-form solution to the line search and there is no need for AFW-L or BoostFW-L. We set $\delta \leftarrow 10^{-7}$ in BoostFW and $\delta \leftarrow 10^{-15}$ in BoostDICG. The results are presented in Figure 4.11.

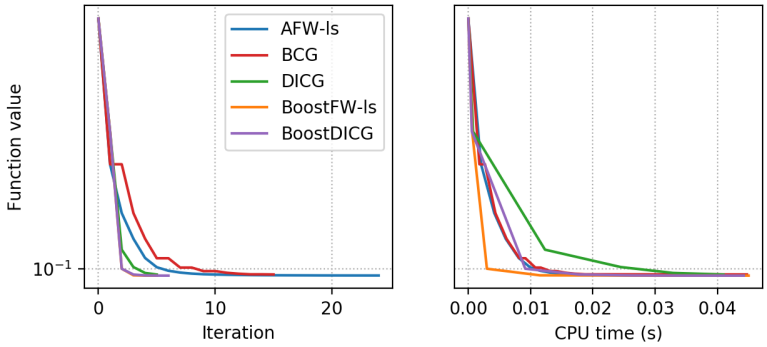

Figure 4.11: Video co-localization on the YouTube-Objects dataset.

All algorithms provide a similar level of performance in function value. In [46], the algorithms are compared with respect to the duality gap on the same experiment. For completeness, we report a similar study in Figure 4.12. The boosting procedure applied to DICG produces very promising empirical results.

Appendix B.2 presents comparisons in duality gap for the other experiments. DICG converges faster than BoostFW in duality gap here (after closing it to 10^{-6} though), but it is not the case in the other experiments.

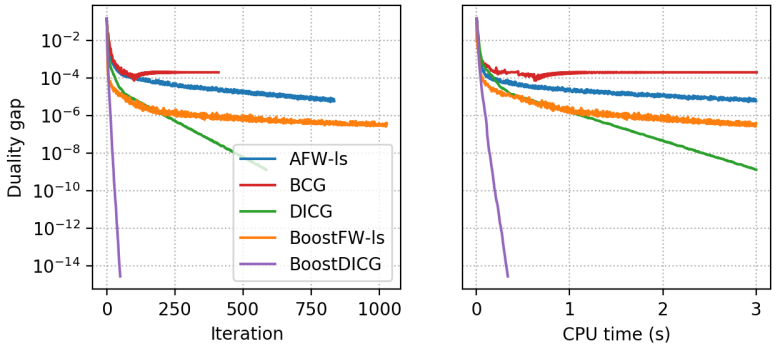

Figure 4.12: Video co-localization on the YouTube-Objects dataset.

4.5 Final remarks

We have proposed an intuitive method to speed up the Frank-Wolfe algorithm by descending in directions better aligned with those of the negative gradients $-\nabla f(x_t)$, all the while remaining projection-free. Our method does not need to maintain the decomposition of the iterates and can naturally be used to boost the performance of any Frank-Wolfe-style algorithm. Although the linear minimization oracle may be called multiple times per iteration, the progress obtained greatly overcomes this cost and leads to strong gains in performance. We demonstrated in a variety of experiments the computational advantage of our method both per iteration and in CPU time over the state of the art. Furthermore, it does not require line search to produce strong performance in practice, which is particularly useful on instances where these are excessively expensive.

Future work may replace the gradient pursuit procedure with a faster conic optimization algorithm to potentially reduce the number of oracle calls. It could also be interesting to investigate how to make each oracle call cheaper via, e.g., *lazification* [15] or subsampling [69].

CHAPTER 5

FRANK-WOLFE WITH ADAPTIVE GRADIENTS FOR LARGE-SCALE OPTIMIZATION

The complexity in large-scale optimization can lie in both handling the objective function and handling the constraint set. In this respect, stochastic Frank-Wolfe algorithms occupy a unique position as they alleviate both computational burdens, by querying only approximate first-order information from the objective and by maintaining feasibility of the iterates without using projections. In this chapter, we improve the quality of their first-order information by blending in adaptive gradients. We derive convergence rates and demonstrate the computational advantage of our method over the state-of-the-art stochastic Frank-Wolfe algorithms on both convex and nonconvex objectives. The experiments further show that our method can improve the performance of adaptive gradient algorithms for constrained optimization.

Based on [26].

5.1 The large-scale optimization setting

We address the constrained finite-sum optimization problem

$$\min_{x \in \mathcal{C}} \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(x) \right\}, \tag{5.1}$$

where $\mathcal{C} \subset \mathbb{R}^n$ is a compact convex set and $f_1, \ldots, f_m \colon \mathbb{R}^n \to \mathbb{R}$ are smooth functions. This setting is quite ubiquitous in machine learning, as many learning problems are formulated via empirical risk minimization [126] and result in the optimization problem (5.1). For example, if $(a_1, y_1), \ldots, (a_m, y_m) \in \mathbb{R}^n \times \mathbb{R}$ denote the set of observed feature-label pairs, then:

(i) the least-squares approach to linear regression puts $f_i(x) = (y_i - \langle a_i, x \rangle)^2$;

(ii) in logistic regression, maximum likelihood estimation with $y_i \in \{-1, 1\}$ leads to $f_i(x) = \ln(1 + \exp(-y_i \langle a_i, x \rangle))$;

(iii) in support vector classification, the smoothed loss can be written $f_i(x) = \max\{0, 1 - y_i \langle a_i, x \rangle\}^2$ to allow soft margins, where again $y_i \in \{-1, 1\}$.

We are interested in developing methods for when the dataset is *large scale*. In this situation, evaluations of the objective f are very expensive and first-order methods cannot be used as they stand. A popular option is to replace, e.g., the evaluation of the gradient with an approximate but much cheaper estimation $\tilde{\nabla} f(x) \leftarrow \nabla f_i(x)$, where i is sampled uniformly at random from $[\![1, m]\!]$. This stochastic approximation method can be traced back to work on Markov chains [119] and works well for problem (5.1). In the context of Frank-Wolfe algorithms, the Stochastic Frank-Wolfe algorithm (SFW) uses an estimator $\tilde{\nabla} f(x_t) \leftarrow (1/b_t) \sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$, where $i_1, \ldots, i_{b_t}$ are sampled i.i.d. uniformly at random from $[\![1, m]\!]$. When the batch-sizes scale as $b_t = \Theta(t^2)$, SFW converges with rate $\mathcal{O}(1/t)$ [58].

Many variants have been proposed to improve the practical efficiency of SFW, also converging at a rate of $\mathcal{O}(1/t)$. From a theoretical standpoint, a popular measure of efficiency in large-scale optimization is the number of gradient evaluations required to achieve ε-convergence. To this end, the Stochastic Variance-Reduced Frank-Wolfe algorithm (SVRF) [58] integrates variance reduction [65, 137] in the estimation of the stochastic gradients to improve the batch-size rate to $b_t = \Theta(t)$. The STOchastic variance-Reduced Conditional gradient sliding algorithm (STORC) [58] builds on the Conditional Gradient Sliding algorithm [82] and further reduces the total number of gradient evaluations by half an order of magnitude, although STORC is not as competitive as SVRF in practice [58, Sec. 5]. This may be because SVRF obtains more progress per gradient evaluation or because the analysis of STORC is more precise. When the objective is additively separable in the data samples, [105] present a Constant batch-size Stochastic Frank-Wolfe algorithm (CSFW) where the batch-sizes do not need to grow over time, i.e., $b_t = \Theta(1)$; in practice, they set

$b_t \leftarrow \lfloor m/100 \rfloor$.

Hence, the number of gradient evaluations required to achieve convergence, although appealing for its theoretical insight, may not necessarily reflect the relative performances of different algorithms in practice. Furthermore, focusing solely on reducing this quantity may actually be ineffective [35]. In this chapter, we take a different route and leverage recent advances in optimization to improve the performance of SFW (and variants) via a better use of first-order information. To the best of our knowledge, it has not yet been explored how to take advantage of adaptive gradients [37, 97], which have been very successful in modern large-scale learning (see, e.g., [34]), in projection-free optimization. Adaptive gradient algorithms consist in setting entrywise step-sizes based on first-order information from past iterates. An interpretation of the success of these methods is that they provide a feature-specific learning rate, which is particularly useful when informative features from the dataset are present in the form of rare events. From an optimization standpoint, these adaptive step-sizes fit better to the loss landscape and alleviate the struggle with ill-conditioning, without requiring access to second-order information.

Contributions. We show that the blend of adaptive gradients and Frank-Wolfe algorithms is successful. We propose a generic template for improving the performance of stochastic Frank-Wolfe algorithms, which applies to all the aforementioned variants. Our method consists in solving the non-Euclidean projection subproblems occurring in the adaptive gradient algorithms *very* incompletely via a fixed and small number of K iterations of the Frank-Wolfe algorithm. We establish convergence guarantees for different implementations of our method, and we demonstrate its computational advantage over the state-of-the-art stochastic Frank-Wolfe algorithms on both convex and nonconvex objectives. Furthermore, the experiments also show that our method can improve the performance of adaptive gradient algorithms on constrained optimization problems. While adaptive gradient algorithms require a non-Euclidean projection at each iteration, our method is projection-free

and still leverages adaptive gradients.

Outline. We start with background materials on stochastic Frank-Wolfe and adaptive gradient algorithms (Sections 5.2–5.3). In Section 5.4, we motivate our approach and present our method through a generic template. We further propose specific implementations and analyze their respective convergence properties. We end that section with practical recommendations and we report computational experiments in Section 5.5. We conclude the chapter with some final remarks in Section 5.6. Appendix B.3 contains an analysis of the sensitivity to K.

5.2 Stochastic Frank-Wolfe algorithms

We present in Template 5.1 the generic template for stochastic Frank-Wolfe algorithms. When $\tilde{\nabla} f(x_t) \leftarrow \nabla f(x_t)$, we obtain the original Frank-Wolfe algorithm (FW).

Template 5.1 Stochastic Frank-Wolfe

Input: Start point $x_0 \in \mathcal{C}$, step-size strategy $(\gamma_t)_{t \in \mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: Update the gradient estimator $\tilde{\nabla} f(x_t)$
3: $v_t \leftarrow \arg\min_{v \in \mathcal{C}} \langle v, \tilde{\nabla} f(x_t) \rangle$
4: $x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
5: **end for**

In order to fit FW to the large-scale finite-sum setting of Problem (5.1), many *stochastic* Frank-Wolfe algorithms have been developed. Most of them follow Template 5.1 and differ only in how they update the gradient estimator $\tilde{\nabla} f(x_t)$ (Line 2). In Table 5.1, we report the strategies adopted in the Stochastic Frank-Wolfe algorithm (SFW) [58], the Stochastic Variance-Reduced Frank-Wolfe algorithm (SVRF) [58], the Stochastic Path-Integrated Differential EstimatoR Frank-Wolfe algorithm (SPIDER-FW) [134, 123], the Online stochastic Recursive Gradient-based Frank-Wolfe algorithm (ORGFW) [132], and the Constant batch-size Stochastic Frank-Wolfe algorithm (CSFW) [105]. SFW is the natural extension

Table 5.1: Gradient estimator updates in stochastic Frank-Wolfe algorithms. The indices $i_1, \ldots, i_{b_t}$ are sampled i.i.d. uniformly at random from $[\![1, m]\!]$. When introduced, $\tilde{x}_t$ denotes the last snapshot iterate and ρ_t denotes the time-varying momentum parameter. CSFW assumes separability of f as $f(x) = (1/m)\sum_{i=1}^{m} f_i(\langle a_i, x\rangle)$.

Algorithm	**Update $\tilde{\nabla} f(x_t)$ in Line 2**
SFW	$\dfrac{1}{b_t}\displaystyle\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$
SVRF	$\nabla f(\tilde{x}_t) + \dfrac{1}{b_t}\displaystyle\sum_{j=1}^{b_t} (\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$
SPIDER-FW	$\nabla f(\tilde{x}_t) + \dfrac{1}{b_t}\displaystyle\sum_{j=1}^{b_t} (\nabla f_{i_j}(x_t) - \nabla f_{i_j}(x_{t-1}))$
ORGFW	$\dfrac{1}{b_t}\displaystyle\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t) + (1-\rho_t)\left(\tilde{\nabla} f(x_{t-1}) - \dfrac{1}{b_t}\displaystyle\sum_{j=1}^{b_t} \nabla f_{i_j}(x_{t-1})\right)$
CSFW	$\tilde{\nabla} f(x_{t-1}) + \displaystyle\sum_{j=1}^{b_t}\left(\frac{1}{m} f_{i_j}'(\langle a_i, x_t\rangle) - [\alpha_{t-1}]_i\right) a_i$
	and $[\alpha_t]_i \leftarrow \begin{cases} (1/m) f_i'(\langle a_i, x_t\rangle) & \text{if } i \in \{i_1, \ldots, i_{b_t}\} \\ [\alpha_{t-1}]_i & \text{else} \end{cases}$

of FW to the large-scale setting of Problem (5.1), SVRF and SPIDER-FW integrate variance reduction based on the works of [65] and [41] respectively, ORGFW uses a form of momentum inspired by [30], and CSFW takes advantage of the additive separability of the objective function in the data samples, when applicable, following the design of [121].

5.3 The Adaptive Gradient algorithm

The Adaptive Gradient algorithm (AdaGrad) [37] (see also [97]) is presented in Algorithm 5.2.

All operations in Line 3 are entrywise in $\mathbb{R}^n$. The matrix $H_t \in \mathbb{R}^{n\times n}$ is diagonal and

Algorithm 5.2 Adaptive Gradient (AdaGrad)

Input: Start point $x_0 \in \mathcal{C}$, offset $\delta > 0$, learning rate $\eta > 0$.

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Update the gradient estimator $\tilde{\nabla} f(x_t)$

3: $H_t \leftarrow \operatorname{diag}\left(\delta 1 + \sqrt{\sum_{s=0}^{t} \tilde{\nabla} f(x_s)^2}\right)$

4: $x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2$

5: **end for**

satisfies for all $i, j \in [\![1, n]\!]$,

$$[H_t]_{i,j} = \begin{cases} \delta + \sqrt{\sum_{s=0}^{t} [\tilde{\nabla} f(x_s)]_i^2} & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{5.2}$$

The default value for the offset hyperparameter is $\delta \leftarrow 10^{-8}$. The new iterate x_{t+1} is computed in Line 4 by solving a constrained convex quadratic minimization subproblem. By completing the square, the subproblem in Line 4 is equivalent to a projection in the metric $\|\cdot\|_{H_t}$:

$$x_{t+1} \leftarrow \arg\min_{x \in \mathcal{C}} \|x - (x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t))\|_{H_t}. \tag{5.3}$$

Ignoring the constraint set $\mathcal{C}$ for ease of exposition, we obtain

$$x_{t+1} \leftarrow x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t),$$

i.e., for every feature $i \in [\![1, n]\!]$,

$$[x_{t+1}]_i \leftarrow [x_t]_i - \frac{\eta [\tilde{\nabla} f(x_t)]_i}{\delta + \sqrt{\sum_{s=0}^{t} [\tilde{\nabla} f(x_s)]_i^2}}. \tag{5.4}$$

Thus, the offset δ prevents from dividing by zero, and we can see that the step-size auto-

matically scales with the geometry of the problem. In particular, infrequent features receive large step-sizes whenever they appear, allowing the algorithm to notice these rare but potentially very informative features.

The family of adaptive gradient algorithms originated with AdaGrad and expanded with RMSProp [125], AdaDelta [136], Adam [71], AMSGrad [117], and, e.g., AdaBound [93, 70], with each new variant addressing some flaws in the previous ones: vanishing step-sizes, incomplete theory, generalization performance [130], etc. For example, RMSProp uses an exponential moving average instead of a sum in Line 3 in order to avoid vanishing step-sizes, since the sum in the denominator of (5.4) can grow too fast for features with dense gradients.

5.4 Frank-Wolfe with adaptive gradients

5.4.1 Our approach

When minimizing an objective over a constraint set, each iteration of AdaGrad can be relatively expensive as it needs to solve the subproblem

$$\min_{x \in \mathcal{C}} \eta \langle x, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2} \|x - x_t\|_{H_t}^2, \tag{5.5}$$

given in Line 4. By (5.3), this is equivalent to a non-Euclidean projection of the unconstrained step $x_t - \eta H_t^{-1} \tilde{\nabla} f(x_t)$. Thus, we could reduce the complexity of AdaGrad by avoiding this projection and moving in the direction of $\arg\min_{v \in \mathcal{C}} \langle G_t, v \rangle$, where $-G_t = -H_t^{-1} \tilde{\nabla} f(x_t)$ denotes the unconstrained descent direction of AdaGrad, as was done in FW for gradient descent with $-G_t = -\nabla f(x_t)$. However, by doing so we may lose the precious properties of the descent directions of AdaGrad, as the directions returned by $\arg\min_{v \in \mathcal{C}} \langle G_t, v \rangle$ can be significantly different from $-G_t$ (Chapter 4).

Thus, instead of avoiding the subproblem (5.5), we can consider solving it incompletely and via a projection-free algorithm. Following [82], at each iteration we could use FW to

solve (5.5) until some specified accuracy ϕ_t is reached, which we check via the duality gap $\max_{v \in \mathcal{C}} \langle \eta \tilde{\nabla} f(x_t) + H_t(x - x_t), v \rangle$. The solution to this procedure would then constitute the new iterate x_{t+1}. The subproblem (5.5) is easy to address since the objective is a simple convex quadratic function, so we can evaluate its exact gradient cheaply and derive the optimal step-size in any descent direction.

However, in order to provide nice theoretical analyses, the sequence of accuracies $(\phi_t)_{t \in \mathbb{N}}$ would need to decay to zero relatively fast, which means that we are back to solving the subproblems completely. This is very time-consuming and overkill in practice. Therefore, instead we propose to perform a fixed number of K iterations on the subproblems, where K is chosen small, e.g., $K \sim 5$. Hence, we choose to leverage just a small amount of information from the adaptive metric H_t, and claim that this will be enough in practice.

5.4.2 The algorithm

We now present our method via a generic template in Template 5.3. We allow the matrix H_t to be relatively general, the only requirements being that it is diagonal and that its entries are clipped to some hand-designed values $[\lambda_t^-, \lambda_t^+]$, as done in [93]. Hence, we can apply the AdaGrad update (5.2), but we can also apply any other variant. Lines 4–10 apply K iterations of FW on

$$\min_{x \in \mathcal{C}} \left\{ Q_t(x) = f(x_t) + \langle x - x_t, \tilde{\nabla} f(x_t) \rangle + \frac{1}{2\eta_t} \|x - x_t\|_{H_t}^2 \right\}. \tag{5.6}$$

This is exactly the subproblem (5.5) with a time-varying learning rate $\eta_t > 0$. We denote by $y_k^{(t)}$ for $k \in [\![0, K]\!]$ the iterates on the subproblem (5.6), starting from $y_0^{(t)} \leftarrow x_t$ (Line 4) and ending at $x_{t+1} \leftarrow y_K^{(t)}$ (Line 11). The step-size $\gamma_k^{(t)}$ in Line 8 is optimal in the sense that $\gamma_k^{(t)} = \arg\min_{\gamma \in [0, \gamma_t]} Q_t(y_k^{(t)} + \gamma(v_k^{(t)} - y_k^{(t)}))$ (Lemma 5.1), where the upper bound γ_t ensures convergence of the sequence $(x_t)_{t \in \mathbb{N}}$.

Template 5.3 Frank-Wolfe with adaptive gradients

Input: Start point $x_0 \in \mathcal{C}$, bounds $0 < \lambda_t^- \leqslant \lambda_{t+1}^- \leqslant \lambda_{t+1}^+ \leqslant \lambda_t^+$ for $t \in \mathbb{N}$, number of inner iterations $K \in \mathbb{N}\backslash\{0\}$, learning rate strategy $(\eta_t)_{t\in\mathbb{N}} \subset \mathbb{R}_{++}$, step-size bounds strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0,1]$.

for $t = 0$ **to** $T-1$ **do**
  Update the gradient estimator $\tilde{\nabla} f(x_t)$
  Update the diagonal matrix H_t and clip its entries to $[\lambda_t^-, \lambda_t^+]$
  $y_0^{(t)} \leftarrow x_t$
  for $k = 0$ **to** $K-1$ **do**
    $\nabla Q_t(y_k^{(t)}) \leftarrow \tilde{\nabla} f(x_t) + \dfrac{1}{\eta_t} H_t(y_k^{(t)} - x_t)$
    $v_k^{(t)} \leftarrow \arg\min_{v\in\mathcal{C}} \langle v, \nabla Q_t(y_k^{(t)})\rangle$
    $\gamma_k^{(t)} \leftarrow \min\left\{\eta_t \dfrac{\langle y_k^{(t)} - v_k^{(t)}, \nabla Q_t(y_k^{(t)})\rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, \gamma_t\right\}$
    $y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
  end for
  $x_{t+1} \leftarrow y_K^{(t)}$
end for

Lemma 5.1. *Consider Template 5.3 and let $t \in \mathbb{N}$. For all $k \in [\![0, K-1]\!]$,*

$$Q_t(y_{k+1}^{(t)}) = \min_{\gamma\in[0,\gamma_t]} Q_t(y_k^{(t)} + \gamma(v_k^{(t)} - y_k^{(t)})).$$

In particular, $Q_t(y_{k+1}^{(t)}) \leqslant Q_t(y_k^{(t)})$ and $Q_t(y_1^{(t)}) \leqslant Q_t(y_0^{(t)} + \gamma_t(v_0^{(t)} - y_0^{(t)}))$.

Proof. Let $k \in [\![0, K-1]\!]$ and $\varphi_k^{(t)} \colon \gamma \in \mathbb{R} \mapsto Q_t(y_k^{(t)} + \gamma(v_k^{(t)} - y_k^{(t)}))$. Then $\varphi_k^{(t)}$ is a convex quadratic function and is minimized at

$$\gamma^* = \eta_t \frac{\langle y_k^{(t)} - v_k^{(t)}, \nabla Q_t(y_k^{(t)})\rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}.$$

Since $v_k^{(t)} \in \arg\min_{v\in\mathcal{C}}\langle v, \nabla Q_t(y_k^{(t)})\rangle$ and $y_k^{(t)} \in \mathcal{C}$, we have $\langle y_k^{(t)} - v_k^{(t)}, \nabla Q_t(y_k^{(t)})\rangle \geqslant 0$ so $\gamma^* \geqslant 0$. Thus, $\varphi_k^{(t)}$ is a decreasing function over $[0, \gamma^*]$. Since $\gamma_k^{(t)} = \min\{\gamma^*, \gamma_t\}$, we

obtain

$$\varphi_k^{(t)}(\gamma_k^{(t)}) = \min_{\gamma \in [0,\gamma_t]} \varphi_k^{(t)}(\gamma),$$

i.e.,

$$Q_t(y_{k+1}^{(t)}) = \min_{\gamma \in [0,\gamma_t]} Q_t(y_k^{(t)} + \gamma(v_k^{(t)} - y_k^{(t)})).$$

□

In Sections 5.4.3–5.4.5, we propose specific implementations of Template 5.3, where gradients are estimated as done in SFW, SVRF, or CSFW (Table 5.1). The diagonal matrices H_t, $t \in \mathbb{N}$, can still be very general. The derived algorithms are named AdaSFW, AdaSVRF, and AdaCSFW respectively, and we analyze their convergence rates. By clipping the entries of H_t, we can control the change rate of the adaptive gradients. However, allowing H_t to be any diagonal matrix with positive entries comes at a price: the upper bound on the convergence rate of the objective is worse than that of the vanilla SFW for example. Nonetheless, the method converges significantly faster in practice (Section 5.5).

Assumption 5.2. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter D and $f_1, \ldots, f_m$ be smooth functions. Let $L = \max_{i \in [\![1,m]\!]} \max_{x,y \in \mathcal{C}} \|\nabla f_i(y) - \nabla f_i(x)\|_2 / \|y - x\|_2$ and $G = \max_{i \in [\![1,m]\!]} \max_{x \in \mathcal{C}} \|\nabla f_i(x)\|_2$.*

Lemma 5.3. *Consider Template 5.3. For all $t \in \mathbb{N}$,*

$$\|x_{t+1} - x_t\|_2 \leqslant KD\gamma_t.$$

Proof. Let $t \in \mathbb{N}$. We have

$$x_{t+1} - x_t = y_K^{(t)} - y_0^{(t)}$$

and, by a straightforward induction on $k \in [\![0, K]\!]$,

$$y_K^{(t)} - y_0^{(t)} = \sum_{k=0}^{K-1} \left(\prod_{\ell=k+1}^{K-1} (1 - \gamma_\ell^{(t)}) \right) \gamma_k^{(t)} (v_k^{(t)} - x_t).$$

Since for all $k \in [\![0, K-1]\!]$,

$$0 \leqslant \gamma_k^{(t)} \leqslant \gamma_t \leqslant 1,$$

we obtain

$$\begin{aligned} \|x_{t+1} - x_t\|_2 &\leqslant \sum_{k=0}^{K-1} \left(\prod_{\ell=k+1}^{K-1} (1 - \gamma_\ell^{(t)}) \right) \gamma_k^{(t)} \|v_k^{(t)} - x_t\|_2 \\ &\leqslant \sum_{k=0}^{K-1} 1 \cdot \gamma_t \cdot D \\ &= K \gamma_t D. \end{aligned}$$

□

5.4.3 SFW with adaptive gradients

We present AdaSFW in Algorithm 5.4. It simply estimates the gradient by averaging over a minibatch.

Algorithm 5.4 AdaSFW

Input: Start point $x_0 \in \mathcal{C}$, batch-size strategy $(b_t)_{t \in \mathbb{N}} \subset \mathbb{N} \backslash \{0\}$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad i_1, \ldots, i_{b_t} \overset{\text{i.i.d.}}{\sim} \mathcal{U}([\![1, m]\!])$
3: $\quad \tilde{\nabla} f(x_t) \leftarrow \dfrac{1}{b_t} \displaystyle\sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$
4: $\quad$ Execute Lines 3–11 of Template 5.3
5: **end for**

Lemma 5.4 is adapted from [58, Apx. B].

Lemma 5.4. *Let Assumption 5.2 hold and consider AdaSFW (Algorithm 5.4). Then for all* $t \in \mathbb{N}$*,*

$$\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2] \leqslant \frac{G}{\sqrt{b_t}}.$$

Proof. Let $t \in \mathbb{N}$. We have $\tilde{\nabla} f(x_t) \leftarrow (1/b_t) \sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)$ so $\mathsf{E}[\tilde{\nabla} f(x_t)] = \nabla f(x_t)$, and

$$\begin{aligned}
\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2^2] &\leqslant \mathsf{E}[\|\tilde{\nabla} f(x_t)\|_2^2] \\
&= \mathsf{E}\left[\left\|\frac{1}{b_t} \sum_{j=1}^{b_t} \nabla f_{i_j}(x_t)\right\|_2^2\right] \\
&= \frac{1}{b_t^2} \sum_{j=1}^{b_t} \mathsf{E}[\|\nabla f_{i_j}(x_t)\|_2^2] \\
&\leqslant \frac{G^2}{b_t},
\end{aligned}$$

where we used that $i_1, \ldots, i_{b_t}$ are sampled i.i.d. in the last two steps. □

Theorem 5.5. *In addition to Assumption 5.2, suppose that* $f_1, \ldots, f_m$ *are convex. Consider AdaSFW (Algorithm 5.4) with* $b_t \leftarrow (G(t+2)/(LD))^2$*,* $\eta_t \leftarrow \lambda_t^-/L$*, and* $\gamma_t \leftarrow 2/(t+2)$ *for all* $t \in \mathbb{N}$*, and let* $\kappa = \lambda_0^+/\lambda_0^-$*. Then for all* $t \geqslant 1$*,*

$$\mathsf{E}[f(x_t)] - \min_{\mathcal{C}} f \leqslant \frac{2LD^2(K+1+\kappa)}{t+1}.$$

Proof. Let $t \in \mathbb{N}$. We have

$$\lambda_{\min}(H)\|\cdot\|_2^2 \leqslant \|\cdot\|_H^2 \leqslant \lambda_{\max}(H)\|\cdot\|_2^2, \tag{5.7}$$

so

$$\frac{L}{2}\|\cdot\|_2^2 \leqslant \frac{L}{2\lambda_t^-}\|\cdot\|_{H_t}^2 = \frac{1}{2\eta_t}\|\cdot\|_{H_t}^2 \tag{5.8}$$

and

$$\frac{1}{2\eta_t}\|\cdot\|_{H_t}^2 \leqslant \frac{\lambda_t^+}{2\eta_t}\|\cdot\|_2^2 = \frac{L}{2}\frac{\lambda_t^+}{\lambda_t^-}\|\cdot\|_2^2 \leqslant \frac{L\kappa}{2}\|\cdot\|_2^2. \tag{5.9}$$

By smoothness of f and (5.8),

$$\begin{aligned}
f(x_{t+1}) &\leqslant f(x_t) + \langle x_{t+1} - x_t, \nabla f(x_t)\rangle + \frac{L}{2}\|x_{t+1} - x_t\|_2^2 \\
&\leqslant f(x_t) + \langle x_{t+1} - x_t, \nabla f(x_t)\rangle + \frac{1}{2\eta_t}\|x_{t+1} - x_t\|_{H_t}^2 \\
&= f(x_t) + \langle x_{t+1} - x_t, \nabla f(x_t)\rangle + \frac{1}{2\eta_t}\|x_{t+1} - x_t\|_{H_t}^2 + \langle x_{t+1} - x_t, \nabla f(x_t) - \tilde{\nabla} f(x_t)\rangle \\
&= Q_t(x_{t+1}) + \langle x_{t+1} - x_t, \nabla f(x_t) - \tilde{\nabla} f(x_t)\rangle \\
&= Q_t(y_K^{(t)}) + \langle x_{t+1} - x_t, \nabla f(x_t) - \tilde{\nabla} f(x_t)\rangle \\
&\leqslant Q_t(y_0^{(t)} + \gamma_t(v_0^{(t)} - y_0^{(t)})) + \|x_{t+1} - x_t\|_2\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2,
\end{aligned}$$

by Lemma 5.1 and the Cauchy-Schwarz inequality. Recall that $y_0^{(t)} = x_t$ and let $v_t v_0^{(t)}$. Then,

$$\begin{aligned}
f(x_{t+1}) &\leqslant Q_t(x_t + \gamma_t(v_t - x_t)) + \|x_{t+1} - x_t\|_2\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2 \\
&= f(x_t) + \gamma_t\langle v_t - x_t, \tilde{\nabla} f(x_t)\rangle + \frac{\gamma_t^2}{2\eta_t}\|v_t - x_t\|_{H_t}^2 + \|x_{t+1} - x_t\|_2\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2.
\end{aligned} \tag{5.10}$$

Let $x^* \in \arg\min_{\mathcal{C}} f$. Since $\nabla Q_t(y_0^{(t)}) = \tilde{\nabla} f(x_t)$, we have $v_t \in \arg\min_{v\in\mathcal{C}}\langle v, \tilde{\nabla} f(x_t)\rangle$ so

$$\begin{aligned}
\langle v_t - x_t, \tilde{\nabla} f(x_t)\rangle &\leqslant \langle x^* - x_t, \tilde{\nabla} f(x_t)\rangle \\
&= \langle x^* - x_t, \nabla f(x_t)\rangle + \langle x^* - x_t, \tilde{\nabla} f(x_t) - \nabla f(x_t)\rangle \\
&\leqslant f(x^*) - f(x_t) + \|x^* - x_t\|_2\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2,
\end{aligned} \tag{5.11}$$

by convexity of f and the Cauchy-Schwarz inequality. Let $\varepsilon_s = f(x_s) - \min_{\mathcal{C}} f$ for all

$t \in \mathbb{N}$. Combining (5.10) and (5.11), subtracting both sides by $\min_{\mathcal{C}} f$, and taking the expectation, we obtain

$$\begin{aligned}\mathsf{E}[\varepsilon_{t+1}] &\leqslant (1-\gamma_t)\mathsf{E}[\varepsilon_t] + \gamma_t \mathsf{E}[\|x^* - x_t\|_2 \|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2] + \frac{\gamma_t^2}{2\eta_t}\mathsf{E}[\|v_t - x_t\|_{H_t}^2] \\ &\quad + \mathsf{E}[\|x_{t+1} - x_t\|_2 \|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2] \\ &\leqslant (1-\gamma_t)\mathsf{E}[\varepsilon_t] + \gamma_t D \mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2] + \frac{\gamma_t^2}{2} L\kappa D^2 + KD\gamma_t \mathsf{E}[\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2],\end{aligned} \tag{5.12}$$

where we used (5.9) and Lemma 5.3. By Lemma 5.4, and with $b_t = (G(t+2)/(LD))^2$ and $\gamma_t = 2/(t+2)$,

$$\begin{aligned}\mathsf{E}[\varepsilon_{t+1}] &\leqslant (1-\gamma_t)\mathsf{E}[\varepsilon_t] + \gamma_t \frac{G}{\sqrt{b_t}} D + \frac{\gamma_t^2}{2} L\kappa D^2 + \frac{G}{\sqrt{b_t}} KD\gamma_t \\ &= \frac{t}{t+2}\mathsf{E}[\varepsilon_t] + \frac{2}{t+2}\frac{LD}{t+2} D + \frac{2}{(t+2)^2} L\kappa D^2 + \frac{LD}{t+2}\frac{2KD}{t+2} \\ &= \frac{t}{t+2}\mathsf{E}[\varepsilon_t] + \frac{C}{(t+2)^2},\end{aligned}$$

where $C = 2LD^2(K+1+\kappa)$. Thus,

$$(t+1)(t+2)\mathsf{E}[\varepsilon_{t+1}] \leqslant t(t+1)\mathsf{E}[\varepsilon_t] + \frac{C(t+1)}{t+2},$$

so, by telescoping,

$$\begin{aligned}t(t+1)\mathsf{E}[\varepsilon_t] &\leqslant 0 \cdot 1 \cdot \mathsf{E}[\varepsilon_0] + \sum_{s=0}^{t-1} \frac{C(s+1)}{s+2} \\ &\leqslant Ct\end{aligned}$$

for all $t \geqslant 1$. Therefore,

$$\mathsf{E}[\varepsilon_t] \leqslant \frac{C}{t+1}$$

for all $t \geqslant 1$. $\square$

Remark 5.6. *Theorem 5.5 simply states that we need to scale the batch-sizes as $b_t = \Theta(t^2)$. We do not need to search for the values of G, L, or D in practice. The same holds for SFW [58].*

We propose in Theorem 5.7 a convergence analysis of AdaSFW on nonconvex objectives. We measure convergence via the duality gap $G\colon x \in \mathcal{C} \mapsto \max_{v\in\mathcal{C}}\langle x - v, \nabla f(x)\rangle$ (Definition 2.1). The duality gap satisfies $G(x) \geqslant 0$, $G(x) = 0$ if and only if x is a stationary point, and, when f is convex, $G(x) \geqslant f(x) - \min_{\mathcal{C}} f$ (Proposition 2.2).

Theorem 5.7. *Let Assumption 5.2 hold and consider AdaSFW (Algorithm 5.4) with $b_t \leftarrow (G/(LD))^2(t+1)$, $\eta_t \leftarrow \lambda_t^-/L$, and $\gamma_t \leftarrow 1/(t+1)^{1/2+\nu}$ where $\nu \in]0, 1/2[$ for all $t \in \mathbb{N}$, and let $\kappa = \lambda_0^+/\lambda_0^-$. For all $t \in \mathbb{N}$, let X_t be sampled uniformly at random from $\{x_0, \dots, x_t\}$. Then for all $t \in \mathbb{N}$,*

$$\mathsf{E}[G(X_t)] \leqslant \frac{(f(x_0) - \min_{\mathcal{C}} f) + LD^2(K + 1 + \kappa/2)\zeta_{1+\nu}}{(t+1)^{1/2-\nu}},$$

where $\zeta_{1+\nu} = \sum_{s=0}^{+\infty} 1/(s+1)^{1+\nu} \in \mathbb{R}_+$. Alternatively, if the time horizon T is fixed, then with $b_t \leftarrow (G/(LD))^2 T$ and $\gamma_t \leftarrow 1/\sqrt{T}$,

$$\mathsf{E}[G(X_{T-1})] \leqslant \frac{(f(x_0) - \min_{\mathcal{C}} f) + LD^2(K + 1 + \kappa/2)}{\sqrt{T}}.$$

Proof. For all $t \in \mathbb{N}$, let E_t denote the conditional expectation with respect to the realization of X_t given $\{x_0, \dots, x_t\}$. Recall that E denotes the expectation with respect to all the randomness in the system. Let $t \in \mathbb{N}$. By (5.10),

$$f(x_{t+1}) \leqslant f(x_t) + \gamma_t\langle v_t - x_t, \tilde{\nabla} f(x_t)\rangle + \frac{\gamma_t^2}{2\eta_t}\|v_t - x_t\|_{H_t}^2 + \|x_{t+1} - x_t\|_2\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2.$$

Let $w_t \in \arg\min_{v\in\mathcal{C}}\langle v, \nabla f(x_t)\rangle$ and note that $G(x_t) = \langle x_t - w_t, \nabla f(x_t)\rangle$. Then, since

$v_t \in \arg\min_{v\in\mathcal{C}}\langle v, \tilde{\nabla} f(x_t)\rangle$,

$$
\begin{aligned}
f(x_{t+1}) &\leqslant f(x_t) + \gamma_t\langle w_t - x_t, \tilde{\nabla} f(x_t)\rangle + \frac{\gamma_t^2}{2\eta_t}\|v_t - x_t\|_{H_t}^2 + \|x_{t+1} - x_t\|_2\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2 \\
&= f(x_t) + \gamma_t\langle w_t - x_t, \nabla f(x_t)\rangle + \gamma_t\langle w_t - x_t, \tilde{\nabla} f(x_t) - \nabla f(x_t)\rangle \\
&\quad + \frac{\gamma_t^2}{2\eta_t}\|v_t - x_t\|_{H_t}^2 + \|x_{t+1} - x_t\|_2\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2 \\
&\leqslant f(x_t) - \gamma_t G(x_t) + \gamma_t D\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2 + \gamma_t^2\frac{L\kappa}{2}D^2 + KD\gamma_t\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2,
\end{aligned}
$$

where we used the Cauchy-Schwarz inequality, (5.9), and Lemma 5.3 in the last inequality. By Lemma 5.4, we obtain

$$
\begin{aligned}
\mathsf{E}[f(x_{t+1})] &\leqslant \mathsf{E}[f(x_t)] - \gamma_t\mathsf{E}[G(x_t)] + \gamma_t\frac{G}{\sqrt{b_t}}D + \gamma_t^2\frac{L\kappa}{2}D^2 + \frac{G}{\sqrt{b_t}}KD\gamma_t \qquad (5.13) \\
&= \mathsf{E}[f(x_t)] - \gamma_t\mathsf{E}[G(x_t)] + \frac{LD^2}{(t+1)^{1+\nu}} + \frac{L\kappa D^2}{2(t+1)^{1+2\nu}} + \frac{KLD^2}{(t+1)^{1+\nu}} \\
&\leqslant \mathsf{E}[f(x_t)] - \gamma_t\mathsf{E}[G(x_t)] + \frac{LD^2(K+1+\kappa/2)}{(t+1)^{1+\nu}},
\end{aligned}
$$

so, by telescoping,

$$
\begin{aligned}
\sum_{s=0}^{t}\gamma_s\mathsf{E}[G(x_s)] &\leqslant \mathsf{E}[f(x_0)] - \mathsf{E}[f(x_{t+1})] + \sum_{s=0}^{t}\frac{LD^2(K+1+\kappa/2)}{(s+1)^{1+\nu}} \\
&\leqslant \Big(f(x_0) - \min_{\mathcal{C}} f\Big) + LD^2(K+1+\kappa/2)\zeta_{1+\nu}.
\end{aligned}
$$

Thus,

$$
\begin{aligned}
\sum_{s=0}^{t}\gamma_s\mathsf{E}[G(x_s)] &\geqslant \gamma_t\sum_{s=0}^{t}\mathsf{E}[G(x_s)] \\
&= \gamma_t(t+1)\mathsf{E}\left[\sum_{s=0}^{t}\frac{1}{t+1}G(x_s)\right] \\
&= (t+1)^{1/2-\nu}\mathsf{E}[\mathsf{E}_t[G(X_t)]] \\
&= (t+1)^{1/2-\nu}\mathsf{E}[G(X_t)],
\end{aligned}
$$

by the law of total expectation. Therefore,

$$\mathsf{E}[G(X_t)] \leqslant \frac{(f(x_0) - \min_{\mathcal{C}} f) + LD^2(K + 1 + \kappa/2)\zeta_{1+\nu}}{(t+1)^{1/2-\nu}}.$$

Alternatively, if the time horizon T is fixed, then by (5.13),

$$\begin{aligned}\mathsf{E}[f(x_{t+1})] &\leqslant \mathsf{E}[f(x_t)] - \frac{1}{\sqrt{T}}\mathsf{E}[G(x_t)] + \frac{LD^2}{T} + \frac{1}{T}\frac{L\kappa}{2}D^2 + \frac{KLD^2}{T} \\ &= \mathsf{E}[f(x_t)] - \frac{1}{\sqrt{T}}\mathsf{E}[G(x_t)] + \frac{LD^2(K + 1 + \kappa/2)}{T},\end{aligned}$$

so, by telescoping,

$$\begin{aligned}\frac{1}{\sqrt{T}}\sum_{t=0}^{T-1}\mathsf{E}[G(x_t)] &\leqslant \mathsf{E}[f(x_0)] - \mathsf{E}[f(x_T)] + LD^2(K + 1 + \kappa/2) \\ &\leqslant \Big(f(x_0) - \min_{\mathcal{C}} f\Big) + LD^2(K + 1 + \kappa/2)\end{aligned}$$

and

$$\begin{aligned}\frac{1}{\sqrt{T}}\sum_{t=0}^{T-1}\mathsf{E}[G(x_t)] &= \frac{T}{\sqrt{T}}\mathsf{E}\left[\sum_{t=0}^{T-1}\frac{1}{T}G(x_t)\right] \\ &= \sqrt{T}\mathsf{E}[\mathsf{E}_{T-1}[G(X_{T-1})]] \\ &= \sqrt{T}\mathsf{E}[G(X_{T-1})],\end{aligned}$$

by the law of total expectation. Therefore,

$$\mathsf{E}[G(X_{T-1})] \leqslant \frac{(f(x_0) - \min_{\mathcal{C}} f) + LD^2(K + 1 + \kappa/2)}{\sqrt{T}}.$$

□

Remark 5.8. *In the first setting of Theorem 5.7, if* $\nu \leftarrow 0.05$ *for example, then* $\mathsf{E}[G(X_t)] = \mathcal{O}(1/t^{0.45})$ *and* $\zeta_{1+\nu} \approx 20.6$.

5.4.4 SVRF with adaptive gradients

We present AdaSVRF in Algorithm 5.5. At every iteration $t = s_k$, $k \in \mathbb{N}$, it computes the exact gradient of the iterate, saves it into memory, then builds the gradient estimator $\tilde{\nabla} f(x_t)$ in the following iterations $t \in [\![s_k + 1, s_{k+1} - 1]\!]$ from this snapshot. Compared to AdaSFW, the variance $\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2^2]$ of the estimator is effectively reduced. The snapshot iterate for x_t is denoted by $\tilde{x}_t$.

Algorithm 5.5 AdaSVRF

Input: Start point $x_0 \in \mathcal{C}$, snapshot strategy $(s_k)_{k\in\mathbb{N}} \subset \mathbb{N}$ such that $s_0 = 0$ and $s_k < s_{k+1}$, batch-size strategy $(b_t)_{t\in\mathbb{N}} \subset \mathbb{N}\backslash\{0\}$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: **if** $t \in \{s_k \mid k \in \mathbb{N}\}$ **then**
3: $\tilde{x}_t \leftarrow x_t$
4: $\tilde{\nabla} f(x_t) \leftarrow \nabla f(\tilde{x}_t)$
5: **else**
6: $\tilde{x}_t \leftarrow \tilde{x}_{t-1}$
7: $i_1, \ldots, i_{b_t} \overset{\text{i.i.d.}}{\sim} \mathcal{U}([\![1, m]\!])$
8: $\tilde{\nabla} f(x_t) \leftarrow \nabla f(\tilde{x}_t) + \frac{1}{b_t} \sum_{j=1}^{b_t} (\nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))$
9: **end if**
10: Execute Lines 3–11 of Template 5.3
11: **end for**

Lemma 5.9 is a slight modification of [58, Lem. 1].

Lemma 5.9. *Let Assumption 5.2 hold and consider AdaSVRF (Algorithm 5.5). For all* $t \in \mathbb{N}$,

$$\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2^2] \leqslant \frac{4L}{b_t} \left(\mathsf{E}\Big[f(x_t) - \min_{\mathcal{C}} f \Big] + \mathsf{E}\Big[f(\tilde{x}_t) - \min_{\mathcal{C}} f \Big] \right).$$

Proof. Let $t \in \mathbb{N}$, E_t denote the conditional expectation with respect to the realization of $i_1, \ldots, i_{b_t}$ given all the randomness in the past (hence, $\tilde{x}_t$ and x_t are given), and $x^* \in$

$\arg\min_{\mathcal{C}} f$. For all $i \in \{i_1, \ldots, i_{b_t}\}$,

$$
\begin{aligned}
&\mathsf{E}_t[\|\nabla f(x_t) - (\nabla f(\tilde{x}_t) + \nabla f_i(x_t) - \nabla f_i(\tilde{x}_t))\|_2^2] \\
&= \mathsf{E}_t[\|\nabla f(x_t) - \nabla f(x^*) + \nabla f(x^*) - \nabla f(\tilde{x}_t) - \nabla f_i(x_t) + \nabla f_i(x^*) - \nabla f_i(x^*) + \nabla f_i(\tilde{x}_t)\|_2^2] \\
&\leqslant 2(\mathsf{E}_t[\|\nabla f(x_t) - \nabla f(x^*) - \nabla f_i(x_t) + \nabla f_i(x^*)\|_2^2] \\
&\quad + \mathsf{E}_t[\|\nabla f(x^*) - \nabla f(\tilde{x}_t) - \nabla f_i(x^*) + \nabla f_i(\tilde{x}_t)\|_2^2]),
\end{aligned}
$$

where we used $(a+b)^2 \leqslant 2(a^2+b^2)$ for all $a, b \in \mathbb{R}$. Since $\mathsf{E}_t[\nabla f_i(x)] = \nabla f(x)$ for all $i \in \{i_1, \ldots, i_{b_t}\}$ and $x \in \{x^*, \tilde{x}_t, x_t\}$, then the first term above is the variance of $\nabla f_i(x_t) - \nabla f_i(x^*)$ and the second term above is the variance of $\nabla f_i(\tilde{x}_t) - \nabla f_i(x^*)$, both with respect to E_t. The variance of a random variable being upper bounded by its second moment, we obtain

$$
\begin{aligned}
&\mathsf{E}_t[\|\nabla f(x_t) - (\nabla f(\tilde{x}_t) + \nabla f_i(x_t) - \nabla f_i(\tilde{x}_t))\|_2^2] \\
&\leqslant 2(\mathsf{E}_t[\|\nabla f_i(x_t) - \nabla f_i(x^*)\|_2^2] + \mathsf{E}_t[\|\nabla f_i(\tilde{x}_t) - \nabla f_i(x^*)\|_2^2]) \\
&\leqslant 4L(\mathsf{E}_t[f_i(x_t) - f_i(x^*) - \langle x_t - x^*, \nabla f_i(x^*)\rangle] \\
&\quad + \mathsf{E}_t[f_i(\tilde{x}_t) - f_i(x^*) - \langle \tilde{x}_t - x^*, \nabla f_i(x^*)\rangle]) \\
&= 4L(\mathsf{E}_t[f_i(x_t)] - \mathsf{E}_t[f_i(x^*)] - \langle x_t - x^*, \mathsf{E}_t[\nabla f_i(x^*)]\rangle \\
&\quad + \mathsf{E}_t[f_i(\tilde{x}_t)] - \mathsf{E}_t[f_i(x^*)] - \langle \tilde{x}_t - x^*, \mathsf{E}_t[\nabla f_i(x^*)]\rangle) \\
&= 4L(f(x_t) - f(x^*) - \langle x_t - x^*, \nabla f(x^*)\rangle \\
&\quad + f(\tilde{x}_t) - f(x^*) - \langle \tilde{x}_t - x^*, \nabla f(x^*)\rangle),
\end{aligned}
$$

by L-smoothness of f_i for all $i \in \{i_1, \ldots, i_{b_t}\}$ and taking the conditional expectation. By convexity of f,

$$
\mathsf{E}_t[\|\nabla f(x_t) - (\nabla f(\tilde{x}_t) + \nabla f_i(x_t) - \nabla f_i(\tilde{x}_t))\|_2^2] \leqslant 4L(f(x_t) - f(x^*) + f(\tilde{x}_t) - f(x^*)).
$$

By the law of total expectation,

$$\begin{aligned}\mathsf{E}[\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2^2] &= \mathsf{E}\left[\left\|\frac{1}{b_t}\sum_{j=1}^{b_t}\left(\nabla f(x_t) - (\nabla f(\tilde{x}_t) + \nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))\right)\right\|_2^2\right] \\ &\leqslant \frac{1}{b_t^2}\sum_{j=1}^{b_t}\mathsf{E}[\|\nabla f(x_t) - (\nabla f(\tilde{x}_t) + \nabla f_{i_j}(x_t) - \nabla f_{i_j}(\tilde{x}_t))\|_2^2] \\ &\leqslant \frac{4L}{b_t}(\mathsf{E}[f(x_t) - f(x^*)] + \mathsf{E}[f(\tilde{x}_t) - f(x^*)]).\end{aligned}$$

□

Theorem 5.10. *In addition to Assumption 5.2, suppose that $f_1, \ldots, f_m$ are convex. Consider AdaSVRF (Algorithm 5.5) with $s_k \leftarrow 2^k - 1$, $b_t \leftarrow 24(K+1+\kappa)(t+2)$ where $\kappa = \lambda_0^+/\lambda_0^-$, $\eta_t \leftarrow \lambda_t^-/L$, and $\gamma_t \leftarrow 2/(t+2)$ for all $t \in \mathbb{N}$. Then for all $t \geqslant 1$,*

$$\mathsf{E}[f(x_t)] - \min_{\mathcal{C}} f \leqslant \frac{2LD^2(K+1+\kappa)}{t+2}. \tag{5.14}$$

Proof. We proceed by strong induction. Let $t \in \mathbb{N}$ and $\varepsilon_s = f(x_s) - \min_{\mathcal{C}} f$ for all $s \in \mathbb{N}$. By (5.12),

$$\mathsf{E}[\varepsilon_{t+1}] \leqslant (1-\gamma_t)\mathsf{E}[\varepsilon_t] + \gamma_t D\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2] + \frac{\gamma_t^2}{2}L\kappa D^2 + KD\gamma_t\mathsf{E}[\|\nabla f(x_t) - \tilde{\nabla} f(x_t)\|_2].$$

If $t = 0$ then, since $s_0 = 0$, we have $\tilde{\nabla} f(x_0) = \nabla f(x_0)$ so

$$\begin{aligned}\mathsf{E}[\varepsilon_1] &\leqslant (1-\gamma_0)\mathsf{E}[\varepsilon_0] + \frac{\gamma_0^2}{2}L\kappa D^2 \\ &= \frac{LD^2\kappa}{2},\end{aligned}$$

because $\gamma_0 = 1$, so the base case holds. Suppose (5.14) holds for all $t' \in [\![1, t]\!]$ for some $t \geqslant 1$. There exist $k, \ell \in \mathbb{N}$ such that $t = s_k + \ell$ and $\ell \leqslant s_{k+1} - s_k - 1$. That is, s_k is the last snapshot time and $\tilde{x}_t = x_{s_k}$. Note that this implies $\ell \leqslant 2^k - 1 = s_k$ so

$t+2 = s_k + \ell + 2 \leqslant 2s_k + 2 \leqslant 2(s_k+2)$. By Lemma 5.9,

$$\begin{aligned}
\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2^2] &\leqslant \frac{4L}{b_t}\left(\mathsf{E}\Big[f(x_t) - \min_{\mathcal{C}} f\Big] + \mathsf{E}\Big[f(\tilde{x}_t) - \min_{\mathcal{C}} f\Big]\right) \\
&\leqslant \frac{4L}{b_t}\left(\frac{2LD^2(K+1+\kappa)}{t+2} + \frac{2LD^2(K+1+\kappa)}{s_k+2}\right) \\
&\leqslant \frac{4L}{b_t}\left(\frac{2LD^2(K+1+\kappa)}{t+2} + \frac{4LD^2(K+1+\kappa)}{t+2}\right) \\
&= \frac{24L^2D^2(K+1+\kappa)}{b_t(t+2)} \\
&= \left(\frac{LD}{t+2}\right)^2,
\end{aligned}$$

since $b_t = 24(K+1+\kappa)(t+2)$. By Jensen's inequality,

$$\begin{aligned}
\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2] &\leqslant \sqrt{\mathsf{E}[\|\tilde{\nabla} f(x_t) - \nabla f(x_t)\|_2^2]} \\
&\leqslant \frac{LD}{t+2}.
\end{aligned}$$

Thus, with $\gamma_t = 2/(t+2)$,

$$\begin{aligned}
\mathsf{E}[\varepsilon_{t+1}] &\leqslant \frac{t}{t+2}\frac{2LD^2(K+1+\kappa)}{t+2} + \frac{2}{t+2}\frac{LD}{t+2}D + \frac{2}{(t+2)^2}L\kappa D^2 + \frac{LD}{t+2}KD\frac{2}{t+2} \\
&= \frac{2LD^2(K+1+\kappa)t + 2LD^2 + 2LD^2\kappa + 2KLD^2}{(t+2)^2} \\
&= \frac{2LD^2(K+1+\kappa)(t+1)}{(t+2)^2} \\
&\leqslant \frac{2LD^2(K+1+\kappa)}{t+3}.
\end{aligned}$$

□

Remark 5.11. *Consider the setting of Theorem 5.10 with the more general strategy* $s_k \leftarrow 2^{k+k_0} - 2^{k_0}$ *and* $b_t \leftarrow 8(2^{k_0+1}+1)(K+1+\kappa)(t+2)$ *where* $k_0 \in \mathbb{N}$. *Then the same result holds. Choosing* $k_0 > 0$ *is useful in practice to avoid computing exact gradients too many times in the early iterations.*

5.4.5 CSFW with adaptive gradients

Here we assume the objective function to be additively separable in the data samples. The problem is

$$\min_{x \in \mathcal{C}} \left\{ f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i(\langle a_i, x \rangle) \right\},$$

where $f_1, \ldots, f_m \colon \mathbb{R} \to \mathbb{R}$ are smooth convex functions and $a_1, \ldots, a_m \in \mathbb{R}^n$ are the data samples. Let $A \in \mathbb{R}^{m \times n}$ be the matrix with rows $a_1^\top, \ldots, a_m^\top$ and $f'(x) = \frac{1}{m} \sum_{i=1}^{m} f_i'(\langle a_i, x \rangle) e_i$ for all $x \in \mathbb{R}^n$. Thus,

$$\nabla f(x) = \frac{1}{m} \sum_{i=1}^{m} f_i'(\langle a_i, x \rangle) a_i = A^\top f'(x). \tag{5.15}$$

We present AdaCSFW in Algorithm 5.6. It estimates the gradient with the quantity $\tilde{\nabla} f(x_t) = \sum_{i=1}^{m} [\alpha_t]_i a_i$ by iteratively updating entries of the vector $\alpha_t \in \mathbb{R}^m$.

Algorithm 5.6 AdaCSFW

Input: Start point $x_0 \in \mathcal{C}$, batch-size strategy $(b_t)_{t \in \mathbb{N}} \subset \mathbb{N} \backslash \{0\}$.

1: $\alpha_{-1} \leftarrow 0 \in \mathbb{R}^m$
2: $\tilde{\nabla} f(x_{-1}) \leftarrow 0 \in \mathbb{R}^n$
3: **for** $t = 0$ **to** $T - 1$ **do**
4: $\quad i_1, \ldots, i_b \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\llbracket 1, m \rrbracket)$
5: $\quad$ **for** $i = 1$ **to** m **do**
6: $\quad\quad$ **if** $i \in \{i_1, \ldots, i_b\}$ **then**
7: $\quad\quad\quad [\alpha_t]_i \leftarrow \frac{1}{m} f_i'(\langle a_i, x_t \rangle)$
8: $\quad\quad$ **else**
9: $\quad\quad\quad [\alpha_t]_i \leftarrow [\alpha_{t-1}]_i$
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: $\quad \tilde{\nabla} f(x_t) \leftarrow \tilde{\nabla} f(x_{t-1}) + \sum_{j=1}^{b} ([\alpha_t]_{i_j} - [\alpha_{t-1}]_{i_j}) a_{i_j}$
13: $\quad$ Execute Lines 3–11 of Template 5.3
14: **end for**

Lemma 5.12 is adapted from [105, Lem. 2 and Lem. 3] and uses Lemma 5.3.

Lemma 5.12. *Consider AdaCSFW (Algorithm 5.6). For all $t \geqslant 1$,*

$$\mathsf{E}_t[\|f'(x_t) - \alpha_t\|_1] \leqslant \left(1 - \frac{b}{m}\right)\left(\|f'(x_{t-1}) - \alpha_{t-1}\|_1 + \frac{KLD_1^A}{m}\gamma_{t-1}\right),$$

where E_t denotes the conditional expectation with respect to the realization of $i_1, \ldots, i_b$ given all the randomness in the past. Thus, for all $t\mathbb{N}$,

$$\mathsf{E}[\|f'(x_t) - \alpha_t\|_1] \leqslant \left(1 - \frac{b}{m}\right)^t \|f'(x_0) - \alpha_0\|_1 + \frac{2KLD_1^A}{m}\left(\left(1 - \frac{b}{m}\right)^{t/2} \ln\left(\frac{t}{2} + 1\right) + \frac{2(m/b - 1)}{t + 2}\right)$$

Proof. Let $t \geqslant 1$ and $i_1, \ldots, i_b$ be the indices sampled at iteration t. For all $i \in [\![1, m]\!]$, we have

$$[\alpha_t]_i = \begin{cases} (1/m) f_i'(\langle a_i, x_t\rangle) & \text{if } i \in \{i_1, \ldots, i_b\}, \\ [\alpha_{t-1}]_i & \text{if } i \notin \{i_1, \ldots, i_b\}. \end{cases}$$

Thus,

$$\begin{aligned} \mathsf{E}_t[\|f'(x_t) - \alpha_t\|_1] &= \sum_{i=1}^m \mathsf{E}_t\left[\left|\frac{1}{m} f_i'(\langle a_i, x_t\rangle - [\alpha_t]_i\right|\right] \\ &= \sum_{i=1}^m \left(1 - \frac{b}{m}\right)\left|\frac{1}{m} f_i'(\langle a_i, x_t\rangle - [\alpha_{t-1}]_i\right| \\ &= \left(1 - \frac{b}{m}\right)\|f'(x_t) - \alpha_{t-1}\|_1. \end{aligned} \tag{5.16}$$

Then, by the triangular inequality and L-smoothness of $f_1, \dots, f_m$,

$$\begin{aligned}
\|f'(x_t) - \alpha_{t-1}\|_1 &\leqslant \|f'(x_t) - f'(x_{t-1})\|_1 + \|f'(x_{t-1}) - \alpha_{t-1}\|_1 \\
&= \sum_{i=1}^{m} \frac{1}{m} |f_i'(\langle a_i, x_t \rangle) - f_i'(\langle a_i, x_{t-1} \rangle)| + \|f'(x_{t-1}) - \alpha_{t-1}\|_1 \\
&\leqslant \frac{L}{m} \sum_{i=1}^{m} |\langle a_i, x_t - x_{t-1} \rangle| + \|f'(x_{t-1}) - \alpha_{t-1}\|_1 \\
&= \frac{L}{m} \|A(x_t - x_{t-1})\|_1 + \|f'(x_{t-1}) - \alpha_{t-1}\|_1. \qquad (5.17)
\end{aligned}$$

Now, similarly to the proof of Lemma 5.3,

$$\begin{aligned}
\|A(x_t - x_{t-1})\|_1 &= \left\| \sum_{k=0}^{K-1} \left(\prod_{\ell=k+1}^{K-1} (1 - \gamma_\ell^{(t-1)}) \right) \gamma_k^{(t-1)} A(v_k^{(t-1)} - x_{t-1}) \right\|_1 \\
&\leqslant \sum_{k=0}^{K-1} 1 \cdot \gamma_{t-1} \cdot \|A(v_k^{(t-1)} - x_{t-1})\|_1 \\
&\leqslant K \gamma_{t-1} D_1^A.
\end{aligned}$$

Together with (5.16) and (5.17), we obtain

$$\mathsf{E}_t[\|f'(x_t) - \alpha_t\|_1] \leqslant \left(1 - \frac{b}{m}\right) \left(\|f'(x_{t-1}) - \alpha_{t-1}\|_1 + \frac{KLD_1^A}{m} \gamma_{t-1} \right).$$

The second result follows as in [105, Lem. 3]. □

Theorem 5.13. *In addition to Assumption 5.2, suppose that $f_1, \dots, f_m$ are convex. Consider AdaCSFW (Algorithm 5.6) with $\eta_t \leftarrow m\lambda_t^- / (L\|A\|_2^2)$ and $\gamma_t \leftarrow 2/(t+2)$ for all $t \in \mathbb{N}$, and let $\kappa = \lambda_0^+ / \lambda_0^-$. Then for all $t \geqslant 1$,*

$$\begin{aligned}
\mathsf{E}[f(x_t)] - \min_{C} f \leqslant \frac{2L}{t+1} &\left(4K(K+1) D_1^A D_\infty^A \left(\frac{1}{b} - \frac{1}{m} \right) + \frac{\kappa \|A\|_2^2 D_2^2}{m} \right) \\
&+ \frac{2(K+1) D_\infty^A (m/b)^2}{t(t+1)} \left(\|f'(x_0) - \alpha_0\|_1 + \frac{16KLD_1^A}{b} \right).
\end{aligned}$$

Proof. By (L/m)-smoothness of $\varphi\colon \xi \in \mathbb{R}^m \mapsto (1/m)\sum_{i=1}^m f_i([\xi]_i)$ [105, Prop. 2], we have

$$\varphi(Ax_{t+1}) \leqslant \varphi(Ax_t) + \langle A(x_{t+1}-x_t), \nabla\varphi(x_t)\rangle + \frac{L}{2m}\|A(x_{t+1}-x_t)\|_2^2,$$

i.e.,

$$\begin{aligned}
f(x_{t+1}) &\leqslant f(x_t) + \langle x_{t+1}-x_t, \nabla f(x_t)\rangle + \frac{L}{2m}\|A(x_{t+1}-x_t)\|_2^2 \\
&\leqslant f(x_t) + \langle x_{t+1}-x_t, \nabla f(x_t)\rangle + \frac{L}{2m}\|A\|_2^2\|x_{t+1}-x_t\|_2^2 \\
&\leqslant f(x_t) + \langle x_{t+1}-x_t, \nabla f(x_t)\rangle + \frac{L}{2m}\frac{\|A\|_2^2}{\lambda_t^-}\|x_{t+1}-x_t\|_{H_t}^2 \\
&= f(x_t) + \langle x_{t+1}-x_t, \nabla f(x_t)\rangle + \frac{1}{2\eta_t}\|x_{t+1}-x_t\|_{H_t}^2 \\
&= f(x_t) + \langle x_{t+1}-x_t, \tilde{\nabla} f(x_t)\rangle + \frac{1}{2\eta_t}\|x_{t+1}-x_t\|_{H_t}^2 + \langle x_{t+1}-x_t, \nabla f(x_t) - \tilde{\nabla} f(x_t)\rangle \\
&= Q_t(x_{t+1}) + \langle x_{t+1}-x_t, \nabla f(x_t) - \tilde{\nabla} f(x_t)\rangle
\end{aligned}$$

by definition of $\|A\|_2$ and (5.7). Thus, with $v_t v_0^{(t)}$ and since $x_{t+1} = y_K^{(t)}$, $x_t = y_0^{(t)}$, by Lemma 5.1 we have

$$\begin{aligned}
f(x_{t+1}) &\leqslant Q_t(x_t + \gamma_t(v_t - x_t)) + \langle x_{t+1}-x_t, \nabla f(x_t) - \tilde{\nabla} f(x_t)\rangle \\
&= f(x_t) + \gamma_t\langle v_t - x_t, \tilde{\nabla} f(x_t)\rangle + \frac{\gamma_t^2}{2\eta_t}\|v_t - x_t\|_{H_t}^2 + \langle A(x_{t+1}-x_t), f'(x_t) - \alpha_t\rangle
\end{aligned}$$

by (5.15). By Hölder's inequality,

$$\begin{aligned}
f(x_{t+1}) &\leqslant f(x_t) + \gamma_t\langle x^* - x_t, \tilde{\nabla} f(x_t)\rangle + \frac{\gamma_t^2}{2\eta_t}\lambda_t^+\|v_t - x_t\|_2^2 + \|A(x_{t+1}-x_t)\|_\infty\|f'(x_t) - \alpha_t\|_1 \\
&= f(x_t) + \gamma_t\langle x^* - x_t, \nabla f(x_t)\rangle + \gamma_t\langle x^* - x_t, \tilde{\nabla} f(x_t) - \nabla f(x_t)\rangle \\
&\quad + \gamma_t^2\frac{\lambda_t^+}{\lambda_t^-}\frac{L\|A\|_2^2}{2m}\|v_t - x_t\|_2^2 + \|A(x_{t+1}-x_t)\|_\infty\|f'(x_t) - \alpha_t\|_1,
\end{aligned}$$

so, by Lemma 5.12,

$$\begin{aligned}
\mathsf{E}[\varepsilon_{t+1}] &\leqslant (1-\gamma_t)\mathsf{E}[\varepsilon_t] + \gamma_t D_\infty^A \mathsf{E}[\|f'(x_t)-\alpha_t\|_1] + \gamma_t^2 \frac{\kappa L \|A\|_2^2}{2m} D_2^2 + K D_\infty^A \gamma_t \mathsf{E}[\|f'(x_t)-\alpha_t\|_1] \\
&= (1-\gamma_t)\mathsf{E}[\varepsilon_t] + \gamma_t (K+1) D_\infty^A \mathsf{E}[\|f'(x_t)-\alpha_t\|_1] + \gamma_t^2 \frac{\kappa L \|A\|_2^2}{2m} D_2^2 \\
&\leqslant \frac{t}{t+2}\mathsf{E}[\varepsilon_t] + \frac{2(K+1)D_\infty^A}{t+2}\left(1-\frac{b}{m}\right)^t \|f'(x_0)-\alpha_0\|_1 + \\
&\quad + \frac{4K(K+1)LD_1^A D_\infty^A}{m(t+2)}\left(\left(1-\frac{b}{m}\right)^{t/2}\ln\left(\frac{t}{2}+1\right) + \frac{2(m/b-1)}{t+2}\right) + \frac{2\kappa L\|A\|_2^2 D_2^2}{m(t+2)^2}.
\end{aligned}$$

Thus, multiplying both sides by $(t+1)(t+2)$,

$$\begin{aligned}
(t+1)(t+2)\mathsf{E}[\varepsilon_{t+1}] &\leqslant t(t+1)\mathsf{E}[\varepsilon_t] + 2(K+1)D_\infty^A(t+1)\left(1-\frac{b}{m}\right)^t \|f'(x_0)-\alpha_0\|_1 \\
&\quad + \frac{4K(K+1)LD_1^A D_\infty^A}{m}(t+1)\left(1-\frac{b}{m}\right)^{t/2}\ln\left(\frac{t}{2}+1\right) \\
&\quad + \frac{8K(K+1)LD_1^A D_\infty^A(m/b-1) + 2\kappa L\|A\|_2^2 D_2^2}{m}\frac{t+1}{t+2}.
\end{aligned}$$

Telescoping, we obtain for all $t \geqslant 1$,

$$\begin{aligned}
t(t+1)\mathsf{E}[\varepsilon_t] &\leqslant 0\cdot 1\cdot \mathsf{E}[\varepsilon_0] + 2(K+1)D_\infty^A\|f'(x_0)-\alpha_0\|_1 \sum_{s=0}^{t-1}(s+1)\left(1-\frac{b}{m}\right)^s \\
&\quad + \frac{4K(K+1)LD_1^A D_\infty^A}{m}\sum_{s=0}^{t-1}(s+1)\left(1-\frac{b}{m}\right)^{s/2}\ln\left(\frac{s}{2}+1\right) \\
&\quad + \frac{2L}{m}\left(4K(K+1)D_1^A D_\infty^A\left(\frac{m}{b}-1\right) + \kappa\|A\|_2^2 D_2^2\right)\sum_{s=0}^{t-1}\frac{s+1}{s+2} \\
&\leqslant 2(K+1)D_\infty^A\|f'(x_0)-\alpha_0\|_1\cdot\left(\frac{m}{b}\right)^2 \\
&\quad + \frac{4K(K+1)LD_1^A D_\infty^A}{m}\cdot 8\left(\frac{m}{b}\right)^3 \\
&\quad + \frac{2L}{m}\left(4K(K+1)D_1^A D_\infty^A\left(\frac{m}{b}-1\right) + \kappa\|A\|_2^2 D_2^2\right)t.
\end{aligned}$$

Therefore,

$$\begin{aligned}\mathsf{E}[\varepsilon_t] \leqslant & \frac{2(K+1)D_\infty^A \|f'(x_0) - \alpha_0\|_1 (m/b)^2}{t(t+1)} \\ & + \frac{32K(K+1)LD_1^A D_\infty^A (m/b)^2}{bt(t+1)} \\ & + \frac{2L}{m} \frac{4K(K+1)D_1^A D_\infty^A (m/b-1) + \kappa \|A\|_2^2 D_2^2}{t+1}.\end{aligned}$$

□

5.4.6 Practical recommendations

We end this section with some practical recommendations. Following Remark 5.11, for convex objectives (Section 5.5.1) we set $k_0 \leftarrow 4$ in SVRF and AdaSVRF; for nonconvex objectives (Section 5.5.2), we took snapshots once per epoch. In all variants of Template 5.3, we used the AdaGrad strategy (5.2) for the adaptive metric H_t. We did not need to clip its entries. The offset was set to the default value $\delta \leftarrow 10^{-8}$. We picked a constant value for the learning rate η_t, tuned in the range $\{10^{i/2} \mid i \in \mathbb{Z}\}$ by starting from $\{10^{i/2} \mid i \in \{-2, 0, 2\}\}$ and then narrowing the search space to $\{10^{i/2} \mid i \in \{i_{\text{best}} - 1, i_{\text{best}}, i_{\text{best}} + 1\}\}$, and extending it if the new i_{best} is at an endpoint. We did not need to bound the step-sizes $\gamma_k^{(t)}$, i.e., we set $\gamma_t \leftarrow 1$. Either way, we noticed that the bounds γ_t obtained from the theoretical analyses were not active in our experiments. Lastly, we found $K \sim 5$ to be a good default value in general, as it provides both low complexity and high performance. A sensitivity analysis is available in Appendix B.3.

5.5 Computational experiments

In this section, we conduct a computational study of our proposed method. We compare it to the state-of-the-art stochastic Frank-Wolfe algorithms, SFW, SVRF [58], SPIDER-FW [134, 123], ORGFW [132], and CSFW [105], as well as to the adaptive gradient algorithms

AdaGrad [37, 97] and AMSGrad [117], which are usually applied to unconstrained problems in the literature. We chose AMSGrad as it solves the non-convergence issue of Adam [71]. The goal is to demonstrate:

(i) that our method improves the performance of projection-free algorithms by blending in adaptive gradients, and

(ii) that it can also be viewed as an efficient adaptive gradient method for constrained optimization, by being projection-free.

At each iteration, AdaGrad and AMSGrad both require a non-Euclidean projection onto the constraint set $\mathcal{C}$. When $\mathcal{C}$ is an ℓ_1-ball, we compute the projection as proposed in [37, Sec. 5.2][1]. For both algorithms, the learning rate η is tuned as explained in Section 5.4.6. The code is available at https://github.com/cyrillewcombettes/adasfw.

5.5.1 Convex objectives

We compare the algorithms on three standard convex optimization problems. We apply Template 5.3 to the best performing variant, demonstrating its flexibility and consistent performance. The performance of each algorithm is evaluated via the duality gap $\max_{v \in \mathcal{C}} \langle x_t - v, \nabla f(x_t) \rangle$. When $\min_{\mathcal{C}} f$ is unknown, the duality gap serves as a measure of convergence and as a stopping criterion (Proposition 2.2). For the batch-sizes, we follow the recommendations given by the theoretical analyses of the respective algorithms. In SFW and SVRF, by Remark 5.6 we set $b_t \sim t^2/\sqrt{m}$ and $b_t \sim t$ respectively, making sure b_t does not grow too fast and stays small compared to the full batch-size m. We have $b_t \leftarrow \max\{2^k \mid t+1 \geqslant 2^k, k \in \mathbb{N}\}$ in SPIDER-FW, and $b_t \leftarrow \lfloor m/100 \rfloor$ in ORGFW, CSFW, AdaGrad, and AMSGrad, following [105] for algorithms where the batch-sizes do not need to grow over time.

[1] In [37, Fig. 3], the `if` statement in Line 2 should be on the condition "$\sum_i a_i v_i \leqslant c$" instead of "$\sum_i v_i \leqslant c$".

Support vector classification. We start with a support vector classification experiment from [38, Eq. (4.3.11)]. Since our work only deals with smooth objective functions[2], we smoothen the hinge loss by taking its square, as done in, e.g., [138]. The problem is

$$\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \max\{0, 1 - y_i \langle a_i, x \rangle\}^2$$
$$\text{s.t. } \|x\|_\infty \leqslant \tau,$$

where the data is generated as follows. For every $(i, j) \in [\![1, m]\!] \times [\![1, n]\!]$, let $a_{i,j} = 0$ with probability $1 - 1/j$, else $a_{i,j} = \pm 1$ equiprobably. Thus, the data matrix $A \in \mathbb{R}^{m \times n}$ has significant variability in the frequency of the features. Then let $u \sim \mathcal{U}(\{-1, 1\}^n)$, and $y_i = \operatorname{sign}(\langle a_i, u \rangle)$ with probability 0.95 else $y_i = -\operatorname{sign}(\langle a_i, u \rangle)$. We have $m = 20\,000$, $n = 1\,000$ and $\tau = 1$. We set $K \leftarrow 2$ and $\eta \leftarrow 10^{-3/2}$ in AdaCSFW, $\eta \leftarrow 10^{-1}$ in AdaGrad, and $\eta \leftarrow 10^{-2}$ in AMSGrad. The results are presented in Figure 5.1.

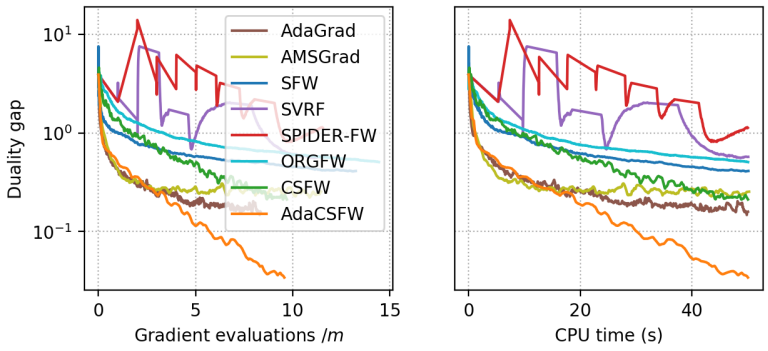

Figure 5.1: Support vector classification on a synthetic dataset.

Linear regression. We consider a linear regression experiment on the YearPrediction-MSD dataset [9], available at https://archive.ics.uci.edu/ml/datasets/YearPredictionMSD. The goal is to predict the release years $y_1, \ldots, y_m$ of songs from their audio features

[2]For Frank-Wolfe on nonsmooth objectives, see [1].

$a_1, \ldots, a_m \in \mathbb{R}^n$. We include a sparsity-inducing constraint via the ℓ_1-norm:

$$\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} (y_i - \langle a_i, x \rangle)^2$$

$$\text{s.t. } \|x\|_1 \leqslant 100.$$

We have $m = 463\,715$ and $n = 90$. We set $K \leftarrow 2$ and $\eta \leftarrow 10^{1/2}$ in AdaSVRF, $\eta \leftarrow 10^{-1/2}$ in AdaGrad, and $\eta \leftarrow 10^{-3/2}$ in AMSGrad. The results are presented in Figure 5.2.

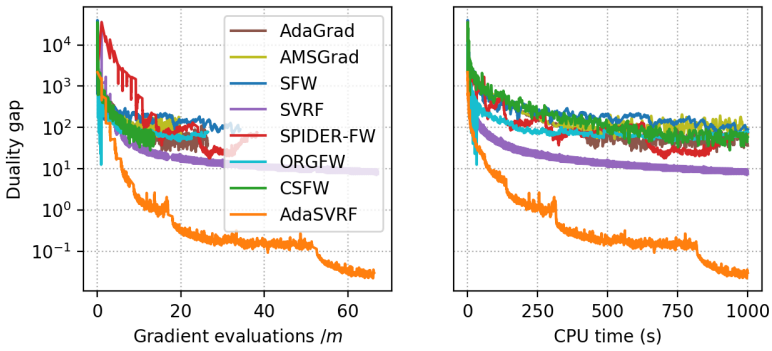

Figure 5.2: Linear regression on the YearPredictionMSD dataset.

Logistic regression. We consider a text categorization experiment on the RCV1 dataset [86]. We use the preprocessed version for binary classification from the LIBSVM library [19], available at https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html#rcv1.binary, and adopt a logistic regression model with a sparsity-inducing constraint via the ℓ_1-norm:

$$\min_{x \in \mathbb{R}^n} \frac{1}{m} \sum_{i=1}^{m} \ln(1 + \exp(-y_i \langle a_i, x \rangle))$$

$$\text{s.t. } \|x\|_1 \leqslant 100,$$

where $y_i \in \{-1, +1\}$. We have $m = 20\,242$ and $n = 47\,236$. We set $K \leftarrow 5$ and $\eta \leftarrow 10^2$ in AdaCSFW, $\eta \leftarrow 1$ in AdaGrad, and $\eta \leftarrow 10^{-1}$ in AMSGrad. The results are presented in Figure 5.3.

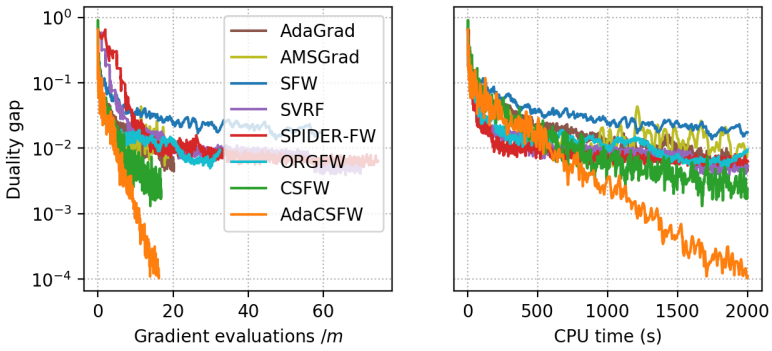

Figure 5.3: Logistic regression on the RCV1 dataset.

5.5.2 Nonconvex objectives

We compare the algorithms on the training of two neural networks. CSFW is not applicable here. Analyses of these algorithms in the nonconvex setting are provided in [118, 134, 132]. Since variance reduction can be ineffective in the training of deep neural networks [35], we run Template 5.3 as AdaSFW only. However, for completeness and transparency, we still compare to the variance-reduced methods, for which we apply *transform locking* as recommended in [35]. We propose a variant of AdaSFW with momentum inspired by Adam and AMSGrad, named AdamSFW; see Appendix A.1. In line with the practice of deep learning, we use constant batch-sizes in all algorithms and every hyperparameter is tuned using the same methodology. Experiments were logged with Weights & Biases [10]. The results are averaged over 5 runs and the shaded areas represent ± 1 standard deviation.

IMDB dataset. We train a neural network for sentiment analysis on the IMDB dataset [95] for 20 epochs. We use the 8 185 subword representation from TensorFlow, available at https://www.tensorflow.org/datasets/catalog/imdb_reviews#imdb_reviewssubwords8k. The neural network has one fully-connected hidden layer of 64 units and ReLU activations. Each layer is constrained into an ℓ_∞-ball with ℓ_2-diameter equal to 6 times the expected ℓ_2-norm of the Glorot uniform initialized values. We set $K \leftarrow 2$ and $\eta \leftarrow 10^{-5/2}$ in AdaSFW, $K \leftarrow 5$ and $\eta \leftarrow 10^{-3}$ in AdamSFW, $\eta \leftarrow 10^{-2}$ in AdaGrad, and $\eta \leftarrow 10^{-7/2}$ in AMSGrad.

The results are presented in Figure 5.4.

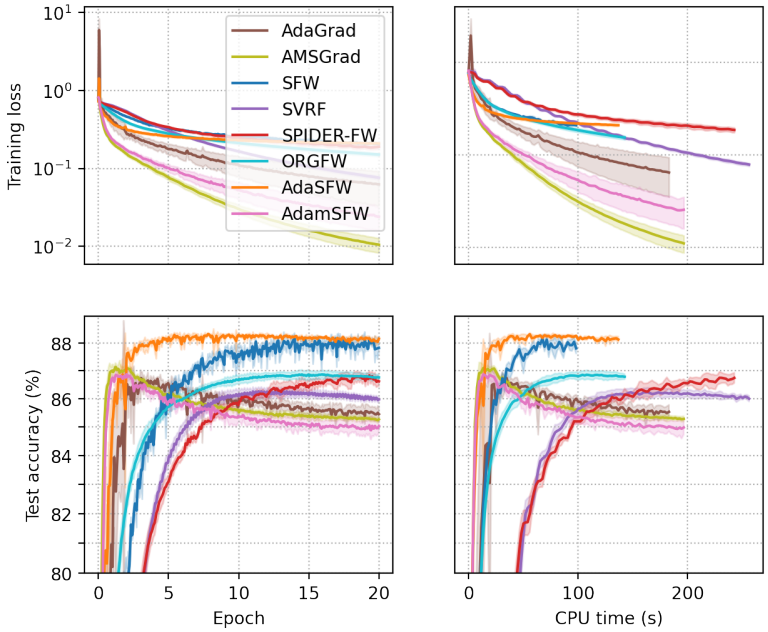

Figure 5.4: Neural network with one fully-connected hidden layer on the IMDB dataset.

AdaSFW provides the best test performance, converging both very fast and to the highest accuracy on the test set, despite optimizing slowly on the training set. The vanilla SFW provides a better test accuracy than its variants SVRF, SPIDER-FW, and ORGFW. AdaGrad, AMSGrad, and AdamSFW optimize very fast on the training set and reach their highest test accuracy early on, which can be favorable if we consider using early stopping.

CIFAR-10 dataset. We train a convolutional neural network (CNN) for image classification on the CIFAR-10 dataset [73], available at https://www.cs.toronto.edu/~kriz/cifar.html, for 100 epochs. It has three 3×3 convolutional layers with 32, 64, and 64 channels respectively, two 2×2 max-pooling layers, one fully-connected hidden layer of 64 units, and ReLU activations. Each layer is constrained into an ℓ_∞-ball with ℓ_2-diameter equal to 200 times the expected ℓ_2-norm of the Glorot uniform initialized values. We set $K \leftarrow 10$ and $\eta \leftarrow 10^{-3/2}$ in AdaSFW, $K \leftarrow 5$ and $\eta \leftarrow 10^{-7/2}$ in AdamSFW, $\eta \leftarrow 10^{-2}$ in AdaGrad, and $\eta \leftarrow 10^{-7/2}$ in AMSGrad. The results are presented in Figure 5.5.

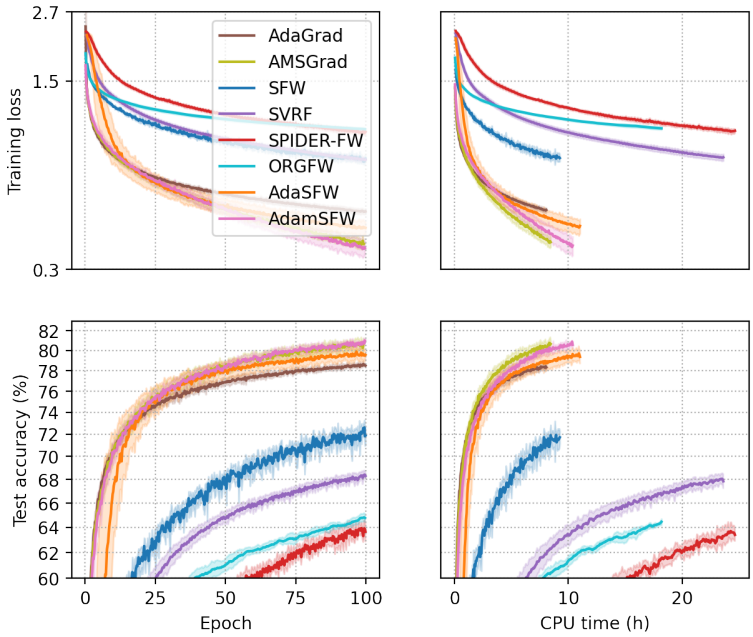

Figure 5.5: Convolutional neural network on the CIFAR-10 dataset.

Here, AdaSFW and AdaGrad, and AdamSFW and AMSGrad, have similar performances respectively. Among projection-free algorithms though, AdaSFW and AdamSFW strongly outperform the other algorithms, while, once again, SVRF, SPIDER-FW, and ORGFW perform worse than the vanilla SFW.

5.6 Final remarks

We have proposed a method for large-scale constrained optimization that augments stochastic Frank-Wolfe algorithms through adaptive gradients. We provided theoretical guarantees and demonstrated its computational advantage over the state-of-the-art stochastic Frank-Wolfe algorithms in a wide range of experiments with both convex and nonconvex objectives. On the training of neural networks, our method is the only projection-free algorithm to improve the performance of the vanilla SFW.

We also demonstrated the computational advantage of our method over adaptive gradient algorithms for constrained optimization. This may be an interesting area for future research as adaptive gradient algorithms have proven successful on a variety of tasks, usually

addressed as unconstrained problems. Furthermore, the computational advantage may be even more pronounced on instances involving contraint sets that are more complex than the ℓ_1-ball and the ℓ_∞-ball, as the non-Euclidean projection required at each iteration in adaptive gradient algorithms can be prohibitively expensive, while our method is projection-free and still leverages adaptive gradients.

CHAPTER 6

FRANK-WOLFE FOR THE APPROXIMATE CARATHÉODORY PROBLEM

The approximate Carathéodory theorem states that given a compact convex set $\mathcal{C} \subset \mathbb{R}^n$ and $p \in [2, +\infty[$, each point $x^* \in \mathcal{C}$ can be approximated to ε-accuracy in the ℓ_p-norm as the convex combination of $\mathcal{O}(pD_p^2/\varepsilon^2)$ vertices of $\mathcal{C}$, where D_p is the diameter of $\mathcal{C}$ in the ℓ_p-norm. A solution satisfying these properties can be built using probabilistic arguments or by applying mirror descent to the dual problem. We revisit the approximate Carathéodory problem by solving the primal problem via the Frank-Wolfe algorithm, providing a simplified analysis and leading to an efficient practical method. Furthermore, improved cardinality bounds are derived naturally using existing convergence rates of the Frank-Wolfe algorithm in different scenarios, when x^* is in the (relative) interior of $\mathcal{C}$, when x^* is the convex combination of a subset of vertices with small diameter, or when $\mathcal{C}$ is uniformly convex. We also propose cardinality bounds when $p \in [1, 2[\cup \{+\infty\}$ via a nonsmooth variant of the algorithm. Lastly, we address the problem of finding sparse approximate projections onto $\mathcal{C}$ in the ℓ_p-norm, $p \in [1, +\infty]$.

Based on [25].

6.1 The approximate Carathéodory problem

Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set and $x^* \in \mathcal{C}$. Suppose that we are interested in expressing x^* as the convex combination of as few vertices of $\mathcal{C}$ as possible. Motivations for this may lie in, e.g., memory space, computation time, or model interpretability. Then Carathéodory's theorem [18] states that this can be achieved with less than $n + 1$ vertices, and this bound is tight. However, in the case where we can afford an ε-approximation in the ℓ_p-norm, where $p \in [1, +\infty]$, can we reduce it to just m points with m being significantly smaller than $n + 1$?

We address the *approximate* Carathéodory problem, which aims at finding a point $x \in \mathcal{C}$ that is the convex combination of a small number of vertices and satisfying $\|x - x^*\|_p \leqslant \varepsilon$. Let the cardinality of x, with respect to a given convex decomposition, be the number of vertices in the decomposition. Informally, we say that x is sparse if it has low cardinality. When $p \in [2, +\infty[$, the approximate Carathéodory theorem states that there exists a solution with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$, where D_p is the diameter of $\mathcal{C}$ in the ℓ_p-norm [3]. The bound is independent of the dimension n, and it is therefore very significant in high-dimensional spaces as it shows that we can obtain extremely sparse solutions. Applications in game theory (Nash equilibria) and combinatorial optimization (densest k-subgraphs) are presented in [3].

The approximate Carathéodory theorem can be proved using Maurey's lemma [113]. A similar proof is presented in [3], which consists in solving the exact Carathéodory problem and then reducing the number of vertices by sampling. A lower bound $\Omega((D_p/\varepsilon)^{p/(p-1)})$ on the cardinality is also provided. Later on, a new proof using only deterministic arguments was proposed in [100], by building the solution via mirror descent [108]. This is particularly useful in practice since the method in [3] is expensive, as solving the exact Carathéodory problem has complexity polynomial in n even when the vertices are known [94][1]. Furthermore, it is proved in [100] that if x^* is in the (relative) interior of $\mathcal{C}$, then a solution with cardinality $\mathcal{O}(p(D_p/r_p)^2 \ln(1/\varepsilon))$ can be found, where $r_p > 0$ denotes the radius of the (affine) ball centered at x^* and contained in $\mathcal{C}$. Finally, they improved the lower bound to $\Omega(pD_p^2/\varepsilon^2)$, thus establishing the optimality of the approximate Carathéodory theorem in the general setting.

When $p = +\infty$, there exists a solution with cardinality $\mathcal{O}(\ln(n)D_\infty^2/\varepsilon^2)$ [3]. When $p \in]1, 2[$, a cardinality bound $\mathcal{O}((1/p)^{1/(p-1)}(D_p/\varepsilon)^{p/(p-1)})$ can be derived from Maurey's lemma; see [13, Lem. D] and [62]. In the more general setting of uniformly smooth Banach spaces, an approximate Carathéodory theorem was recently proposed in [62].

[1] In [94, Thm. 3.1], the dimension of the ambient space is denoted by d.

Table 6.1: Cardinality bounds to achieve ε-convergence in the approximate Carathéodory problem with respect to the ℓ_p-norm. (A1): $x^* \in \operatorname{ri} \mathcal{C}$. (A2): $\mathcal{C}$ is α_p-strongly convex. (A3): $\mathcal{C}$ is (α_p, q_p)-uniformly convex, $q_p \in [2, +\infty[$.

ℓ_p-norm	Assumption	Cardinality bound	
		Ours	Related work
$p \in [2, +\infty[$	–	$\mathcal{O}\left(\frac{pD_p^2}{\varepsilon^2}\right)$ or $\mathcal{O}\left(\frac{p(D_*^2 + D_0^2)}{\varepsilon^2}\right)$ (Corollaries 6.9–6.10)	$\mathcal{O}\left(\frac{pD_p^2}{\varepsilon^2}\right)$ [3, 100, 62]
	(A1)	$\mathcal{O}\left(p\left(\frac{D_p}{r_p}\right)^2 \ln\left(\frac{1}{\varepsilon}\right)\right)$ (Corollary 6.11)	$\mathcal{O}\left(p\left(\frac{D_p}{r_p}\right)^2 \ln\left(\frac{1}{\varepsilon}\right)\right)$ [100]
	(A2)	$\mathcal{O}\left(\frac{\sqrt{p}D_p + p/\alpha_p}{\varepsilon}\right)$ (Corollary 6.12)	–
	(A3)	$\mathcal{O}\left(\frac{(pD_p^2)^{(q_p-1)/q_p} + p/\alpha_p^{2/q_p}}{\varepsilon^{2(q_p-1)/q_p}}\right)$ (Corollary 6.13)	–
$p \in]1, 2[$	–	$\mathcal{O}\left(\frac{n^{(2-p)/p} D_2^2}{\varepsilon^2}\right)$ (Corollaries 6.18 and 6.21)	$\mathcal{O}\left(\frac{D_p^{p/(p-1)}}{p^{1/(p-1)}\varepsilon^{p/(p-1)}}\right)$ [13, 62]
$p = 1$	–	$\mathcal{O}\left(\frac{nD_2^2}{\varepsilon^2}\right)$ (Corollaries 6.18 and 6.21)	–
$p = +\infty$	–	$\mathcal{O}\left(\frac{D_2^2}{\varepsilon^2}\right)$ (Corollaries 6.19 and 6.22)	$\mathcal{O}\left(\frac{\ln(n) D_\infty^2}{\varepsilon^2}\right)$ [3]

Contributions. We address the approximate Carathéodory problem in the ℓ_p-norm via the Frank-Wolfe algorithm (FW). We cover the whole range $p \in [1, +\infty]$, with a slight modification of FW when $p \in [1, 2[\cup \{+\infty\}$. When $p \in [2, +\infty[$, we recover the cardinal-

ity bound $\mathcal{O}(pD_p^2/\varepsilon^2)$ by addressing the primal problem directly. This is in contrast with the approach in [100], which consists of formulating the dual problem and solving it via mirror descent; although it is pointed out that this selects the exact same set of vertices as if FW was applied to the primal problem [2], our analysis is much simpler. Moreover, the method in [100] for the case $x^* \in \operatorname{ri} \mathcal{C}$ requires restarting mirror descent and knowledge of the radius r_p, which may not be available. We show that a direct application of FW generates the desired solution, i.e., that FW is adaptive to the properties of the problem. Our approach further provides improved cardinality bounds when x^* is the convex combination of a subset of vertices with small diameter or when $\mathcal{C}$ is uniformly convex. When $p \in [1, 2[$, we build a solution with cardinality $\mathcal{O}(n^{(2-p)/p} D_2^2/\varepsilon^2)$ via a nonsmooth variant of FW. This improves the dependence to ε in the previous known bound for $p \in]1, 2[$ but involves a term (at most linear) in the dimension n. The nonsmooth FW variant also finds a solution with cardinality $\mathcal{O}(D_2^2/\varepsilon^2)$ when $p = +\infty$, which is dimension-free compared to the result in [3]. Finally, we address the problem of finding sparse approximate projections in the ℓ_p-norm.

Outline. The bulk of the chapter considers the case $p \in [2, +\infty[$. In Section 6.2, we show that the Frank-Wolfe algorithm is an intuitive method to solve the approximate Carathéodory problem. In Section 6.4, we prove that it solves the approximate Carathéodory theorem and that it also generates improved cardinality bounds in different scenarios, using faster convergence rates of the algorithm (Section 6.3). In Section 6.5, we analyze the case $p \in [1, 2[\cup \{+\infty\}$ via a nonsmooth variant of the Frank-Wolfe algorithm. In Section 6.6, we address the problem of finding sparse approximate projections. We present computational experiments in Section 6.7. We briefly mention in Section 6.7.2 that the example of a simple lower bound $\Omega(1/\varepsilon^2)$ using Hadamard matrices in [100] is actually a lower bound $\Omega(1/(\varepsilon^2 + 1/n))$.

6.2 Frank-Wolfe and the approximate Carathéodory problem

Given a compact convex set $\mathcal{C} \subset \mathbb{R}^n$, a point $x^* \in \mathcal{C}$, and an ℓ_p-norm where $p \in [2, +\infty[$, the approximate Carathéodory problem can be formulated as the problem of finding a sparse approximate solution to

$$\min_{x \in \mathcal{C}} \frac{1}{2} \|x - x^*\|_p^2. \tag{6.1}$$

Putting a square on the ℓ_p-norm provides the objective $f \colon x \in \mathbb{R}^n \mapsto (1/2)\|x - x^*\|_p^2$ with several properties favorable to optimization (Lemma 6.1).

Lemma 6.1. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set, $x^* \in \mathcal{C}$, $p \in [2, +\infty[$, and $f \colon x \in \mathbb{R}^n \mapsto (1/2)\|x - x^*\|_p^2$. Then f is convex, $(p-1)$-smooth and 1-gradient dominated on $\mathbb{R}^n$, and $\sqrt{2}$-sharp on $\mathcal{C}$, all respect to the ℓ_p-norm.*

Proof. The convexity and the $\sqrt{2}$-sharpness of f are trivial. Let $h \colon x \in \mathbb{R}^n \mapsto (1/2)\|x\|_p^2$. For all $q \in]1, 2]$, $g \colon y \in \mathbb{R}^n \mapsto (1/2)\|y\|_q^2$ is $(q-1)$-strongly convex with respect to the ℓ_q-norm [122, Lem. 17]. Let $q = p/(p-1) \in]1, 2]$. Then the dual norm of the ℓ_q-norm is the ℓ_p-norm and the conjugate of g is h [40, Rem. I.4.1]. By [135, Cor. 3.5.11 and Rem. 3.5.3], h is $1/(q-1)$-smooth with respect to the ℓ_p-norm, i.e., f is $(p-1)$-smooth with respect to the ℓ_p-norm. Lastly, let $x \in \mathbb{R}^n$. We have

$$\nabla f(x) = \|x - x^*\|_p^{2-p} \begin{pmatrix} \vdots \\ \operatorname{sign}([x - x^*]_i)|[x - x^*]_i|^{p-1} \\ \vdots \end{pmatrix}. \tag{6.2}$$

Thus,

$$\begin{aligned}\|\nabla f(x)\|_q^2 &= \|x - x^*\|_p^{2(2-p)} \left(\sum_{i=1}^n |[x - x^*]_i|^{q(p-1)} \right)^{2/q} \\ &= \|x - x^*\|_p^{2(2-p)} \left(\sum_{i=1}^n |[x - x^*]_i|^{p} \right)^{2(p-1)/p} \\ &= \|x - x^*\|_p^2.\end{aligned}$$

Therefore, f is 1-gradient dominated with respect to the ℓ_p-norm. □

A natural strategy to solve problem (6.1) is to start from an arbitrary vertex $x_0 \in \mathcal{C}$ and to sequentially pick up new vertices until the iterates have converged to the desired accuracy. This is exactly the design of the Frank-Wolfe algorithm. Thus, given a desired level of accuracy $\varepsilon > 0$, we can apply FW to problem (6.1) and count the number of iterations until $\|x_t - x^*\|_p \leqslant \varepsilon$, i.e., until $f(x_t) - f(x^*) \leqslant \varepsilon^2/2$. We can then provide bounds on the cardinality of the solution based on the convergence analysis of FW. In Section 6.3, we study these in different scenarios. In practice, since we know the value of $f(x^*) = 0$, we can observe the primal gap directly and use it at a stopping criterion to actually realize the cardinality bounds.

6.3 Faster convergence rates of the Frank-Wolfe algorithm

In this section, we present faster convergence rates of the Frank-Wolfe algorithm. Throughout, we consider an arbitrary norm $\|\cdot\|$ on $\mathbb{R}^n$.

Assumption 6.2. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter D and $f\colon \mathbb{R}^n \to \mathbb{R}$ be an L-smooth convex function on $\mathcal{C}$, where D and L are defined with respect to $\|\cdot\|$.*

As seen in Section 2.1, under Assumption 6.2, the Frank-Wolfe algorithm (FW, Algo-

rithm 2.1) is a first-order method addressing the constrained convex optimization problem

$$\min_{x \in \mathcal{C}} f(x). \tag{6.3}$$

The general convergence rate of FW on problem (6.3) is $\mathcal{O}(1/t)$ (Theorem 2.6). However, faster rates can be established in particular situations.

6.3.1 Faster convergence rates under additional assumptions

Faster rates of FW can be established with no modification of the algorithm nor of the step-size strategy. This shows that FW is adaptive and naturally leverages the structure of the problem. A summary is provided in Table 6.2. Let $\mathcal{X} = \arg\min_{\mathbb{R}^n} f$ denote the set of unconstrained solutions, possibly empty.

Table 6.2: Additional assumptions and corresponding convergence rates of FW on problem (2.1), where $\mathcal{C}$ is a compact convex set and f is a smooth convex function (Assumption 6.2). We denote by $\mathcal{X} = \arg\min_{\mathbb{R}^n} f$ the set of unconstrained solutions, possibly empty. The strong convexity assumption can be generalized to that of uniform convexity and also leads to faster rates (Theorems 6.6–6.7).

Additional assumptions				**Rate**
$\mathcal{C}$ strongly convex	f gradient dominated	$\mathcal{X} \cap \operatorname{ri} \mathcal{C} \neq \varnothing$	$\mathcal{X} \cap \mathcal{C} = \varnothing$	
✗	✗	✗	✗	$\mathcal{O}(1/t)$
✗	✓	✓	✗	$\mathcal{O}(\exp(-\omega t))$
✓	✗	✗	✓	$\mathcal{O}(\exp(-\omega t))$
✓	✓	✗	✗	$\mathcal{O}(1/t^2)$

If there exists an unconstrained solution $x^* \in \arg\min_{\mathbb{R}^n} f$ in the (relative) interior of $\mathcal{C}$ and if f is gradient dominated, then FW converges at a linear rate, as shown in [45, Sec. 4.2] following an argument similar to [50]. In Theorem 6.3, r is the radius of an (affine) ball centered at x^* and contained in $\mathcal{C}$.

Theorem 6.3. *In addition to Assumption 6.2, suppose that* $\arg\min_{\mathbb{R}^n} f \cap \operatorname{ri} \mathcal{C} \neq \varnothing$ *and* f *is* μ*-gradient dominated on* $\mathcal{C}$ *with respect to* $\|\cdot\|$*. Consider FW (Algorithm 2.1) with the*

closed-loop strategy (2.2). *Then for all* $t \geqslant 1$,

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2}\left(1 - \frac{\mu}{L}\left(\frac{r}{D}\right)^2\right)^{t-1},$$

where $r \in]0, D/4]$.

On the other end, if all unconstrained solutions are outside of $\mathcal{C}$ and if $\mathcal{C}$ is strongly convex, then FW converges again at a linear rate [85]. The distance of $\arg\min_{\mathbb{R}^n} f$ to $\mathcal{C}$ is measured via the quantity $c = \min_{\mathcal{C}} \|\nabla f\|_* > 0$.

Theorem 6.4. *In addition to Assumption 6.2, suppose that* $\mathcal{C}$ *is* α*-strongly convex with respect to* $\|\cdot\|$ *and* $\mathcal{C} \cap \arg\min_{\mathbb{R}^n} f = \varnothing$. *Consider FW (Algorithm 2.1) with the closed-loop strategy* (2.2). *Then for all* $t \geqslant 1$,

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{LD^2}{2}\left(1 - \min\left\{\frac{1}{2}, \frac{\alpha c}{4L}\right\}\right)^{t-1},$$

where $c = \inf_{\mathcal{C}} \|\nabla f\|_* > 0$.

Theorems 6.3–6.4 rely on the location of the set of unconstrained solutions $\arg\min_{\mathbb{R}^n} f$ with respect to $\mathcal{C}$, and the convergence rates become increasingly slower as this set comes closer to the boundary of $\mathcal{C}$, which can be seen with $r \to 0$ and $c \to 0$ respectively. However, if $\mathcal{C}$ is strongly convex and f is gradient dominated, FW enjoys a faster rate independently of the location of $\arg\min_{\mathbb{R}^n} f$ [45].

Theorem 6.5. *In addition to Assumption 6.2, suppose that* $\mathcal{C}$ *is* α*-strongly convex and* f *is* μ*-gradient dominated on* $\mathcal{C}$*, both with respect to* $\|\cdot\|$. *Consider FW (Algorithm 2.1) with the closed-loop strategy* (2.2). *Then for all* $t \geqslant 1$,

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{\max\{(9/2)LD^2, 144(L/\alpha)^2/\mu\}}{(t+2)^2}.$$

The notion of strong convexity for a set can be generalized to that of uniform convexity.

Theorems 6.6–6.7 are slightly adapted from [67] using Lemma 2.5.

Theorem 6.6. *In addition to Assumption 6.2, suppose that $\mathcal{C}$ is (α, q)-uniformly convex with respect to $\|\cdot\|$, where $q > 2$, and $\mathcal{C} \cap \arg\min_{\mathbb{R}^n} f = \varnothing$. Consider FW (Algorithm 2.1) with the closed-loop strategy* (2.2)*. Then for all $t \geqslant 1$,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{\max\{(LD^2/2)(1+\beta_1)^{q/(q-2)}, 4(L/\beta_2)^{q/(q-2)}(4/(\alpha c))^{2/(q-2)}\}}{(t+\beta_1)^{q/(q-2)}},$$

where $\beta_1 = (2 - 2^{(q-2)/q})/(2^{(q-2)/q} - 1)$ and $\beta_2 = (q-2)/q - (2/q)(2^{(q-2)/q} - 1)$.

Theorem 6.7. *In addition to Assumption 6.2, suppose that $\mathcal{C}$ is (α, q)-uniformly convex, where $q \geqslant 2$, and f is σ-sharp on $\mathcal{C}$, both with respect to $\|\cdot\|$. Consider FW (Algorithm 2.1) with the closed-loop strategy* (2.2)*. Then for all $t \geqslant 1$,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{\max\{(LD^2/2)(1+\beta_1)^{q/(q-1)}, 2(L/\beta_2)^{q/(q-1)}(\sigma/\alpha)^{2/(q-1)}\}}{(t+\beta_1)^{q/(q-1)}},$$

where $\beta_1 = (2 - 2^{(q-1)/q})/(2^{(q-1)/q} - 1)$ and $\beta_2 = (q-1)/q - (1/q)(2^{(q-1)/q} - 1)$.

6.3.2 An improved convergence rate with an enhanced oracle

A desired feature for the approximate Carathéodory problem is to have a faster convergence rate for FW when the solutions have a sparse representation. However, the results in Section 6.3.1 do not provide such a result. Denote by $\mathcal{V} \subset \mathcal{C}$ the set of vertices of $\mathcal{C}$. By replacing the linear minimization problem in FW with

$$\min_{v \in \mathcal{V}} f(x_t) + \gamma_t \langle v - x_t, \nabla f(x_t) \rangle + \frac{L}{2}\gamma_t^2 \|v - x_t\|_2^2, \tag{6.4}$$

which quantity appears when applying the smoothness inequality for f between x_t and $x_t + \gamma_t(v - x_t)$, an improvement on the general convergence rate can be obtained [47].

Note that problem (6.4) is constrained to $\mathcal{V}$ instead of $\mathcal{C}$, and can be written

$$\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) \rangle + \lambda_t \|v - x_t\|_2^2, \tag{6.5}$$

where $\lambda_t = L\gamma_t/2$. In many situations, it actually reduces to a linear minimization problem over $\mathcal{C}$, and therefore does not burden the algorithm. For example, if $\mathcal{V} \subset \{0,1\}^n$, then $\|v\|_2^2 = \langle v, 1 \rangle$ for all $v \in \mathcal{V}$ so

$$\begin{aligned}\arg\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) \rangle + \lambda_t \|v - x_t\|_2^2 &= \arg\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) \rangle + \lambda_t \|v\|_2^2 - 2\lambda_t \langle v, x_t \rangle \\ &= \arg\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) + \lambda_t (1 - 2x_t) \rangle,\end{aligned}$$

or, if $\|u\|_2 = \|v\|_2$ for all $u, v \in \mathcal{V}$, then

$$\arg\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) \rangle + \lambda_t \|v - x_t\|_2^2 = \arg\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) - 2\lambda_t x_t \rangle.$$

Since problem (6.5) is equivalent to $\min_{v \in \mathcal{V}} \|v - (x_t - \nabla f(x_t)/(2\lambda_t))\|_2^2$, it is called the *nearest extreme point* (NEP) oracle. The algorithm is presented in Algorithm 6.1, where the smoothness constant L is with respect to the ℓ_2-norm.

Algorithm 6.1 Frank-Wolfe with a Nearest Extreme Point oracle (NEP-FW)

Input: Start point $x_0 \in \mathcal{C}$, smoothness constant $L > 0$, step-size strategy $(\gamma_t)_{t \in \mathbb{N}} \subset [0, 1]$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{V}} \langle v, \nabla f(x_t) \rangle + \frac{L}{2} \gamma_t \|v - x_t\|_2^2$
3: $\quad x_{t+1} \leftarrow x_t + \gamma_t (v_t - x_t)$
4: **end for**

With this modification, NEP-FW improves the bound in the convergence rate of FW [47]. Comparing to Theorem 2.6, the improvement is significant when the solutions lie in the convex hull of a subset of vertices with small diameter and the start point x_0 is of good quality.

Theorem 6.8. *Let Assumption 6.2 hold with respect to the ℓ_2-norm and consider NEP-FW (Algorithm 6.1) with the open-loop strategy* (2.4)*. Then for all $t \geqslant 1$,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{2L(D_*^2 + D_0^2)}{t+2},$$

where $D_* = \min_{\mathcal{S} \subset \mathcal{V}, \arg\min_{\mathcal{C}} f \subset \operatorname{conv} \mathcal{S}} \operatorname{diam} \mathcal{S}$ and $D_0 = \operatorname{diam}\{v \in \mathcal{V} \mid f(v) \leqslant f(x_0)\}$ are defined with respect to the ℓ_2-norm.

6.4 Application to the approximate Carathéodory problem

As explained in Section 6.2, we can obtain a solution to the approximate Carathéodory problem by applying the Frank-Wolfe algorithm to problem (6.1) and the cardinality of the solution can be derived from the convergence analysis in Section 6.3, with Lemma 6.1. We denote by $\mathcal{V} \subset \mathcal{C}$ the set of vertices and by $D_p = \max_{x,y \in \mathcal{C}} \|y - x\|_p$ the diameter of $\mathcal{C}$ in the ℓ_p-norm. Note that here, by Lemma 6.1 and (6.2), the closed-loop strategy (2.2) reads

$$\gamma_t \leftarrow \min \left\{ \frac{\|x_t - x^*\|_p^{2-p} \langle x_t - v_t, \operatorname{sign}(x_t - x^*) |x_t - x^*|^{p-1} \rangle}{(p-1)\|x_t - v_t\|_p^2}, 1 \right\} \tag{6.6}$$

where $\operatorname{sign}(x_t - x^*)|x_t - x^*|^{p-1} = (\operatorname{sign}([x_t - x^*]_i)|[x_t - x^*]_i|^{p-1})_{i \in [\![1,n]\!]} \in \mathbb{R}^n$.

Corollary 6.9 follows from Theorem 2.6 and shows that FW generates a solution with the optimal $\mathcal{O}(pD_p^2/\varepsilon^2)$ number of vertices. Therefore, a solution to the approximate Carathéodory problem in the ℓ_p-norm can be obtained via FW.

Corollary 6.9. *By running FW on problem* (6.1) *with the closed-loop strategy* (6.6) *or the open-loop strategy* (2.4)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$.*

Another possibility is to run NEP-FW. Following Theorem 6.8, Corollary 6.10 shows that we can obtain a better cardinality bound when x^* is the convex combination of a subset of vertices with small diameter and x_0 is a good start.

Corollary 6.10. *By running NEP-FW on problem* (6.1) *with the open-loop strategy* (2.4)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(p(D_*^2 + D_0^2)/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$, where $D_* = \min_{\mathcal{S} \subset \mathcal{V}, x^* \in \operatorname{conv} \mathcal{S}} \operatorname{diam} \mathcal{S}$ and $D_0 = \operatorname{diam}\{v \in \mathcal{V} \mid f(v) \leqslant f(x_0)\}$ are defined with respect to the ℓ_2-norm.*

Proof. The result follows from Theorem 6.8 and Lemma 6.1, because f is also $(p-1)$-smooth with respect to the ℓ_2-norm since $\|\cdot\|_p \leqslant \|\cdot\|_2$ for $p \in [2, +\infty[$. □

It is likely that the point x^* we want to approximate is in the (relative) interior of $\mathcal{C}$. Following Theorem 6.3, Corollary 6.11 improves the cardinality bound in this scenario, to a logarithmic dependence on $1/\varepsilon$. Note that a similar result is obtained in [100], however they assume knowledge of the radius of an (affine) ball centered at x^*, which may not be available. This is not required in FW as the method naturally adapts to the structure of the problem.

Corollary 6.11. *Suppose that $x^* \in \operatorname{ri} \mathcal{C}$ and let r_p be the radius of an affine ball centered at x^* and contained in $\mathcal{C}$, with respect to the ℓ_p-norm. Then by running FW on problem* (6.1) *with the closed-loop strategy* (6.6)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(p(D_p/r_p)^2 \ln(1/\varepsilon))$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$.*

Another scenario is when $\mathcal{C}$ has a particular shape. Following Theorem 6.5, Corollary 6.12 shows an improved cardinality bound when $\mathcal{C}$ is strongly convex. It is actually subsumed by Corollary 6.13, which follows from Theorem 6.7. Note that $2(q_p - 1)/q_p \in [1, 2[$ for $q_p \in [2, +\infty[$.

Corollary 6.12. *Suppose that $\mathcal{C}$ is α_p-strongly convex with respect to the ℓ_p-norm. Then by running FW on problem* (6.1) *with the closed-loop strategy* (6.6)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}((\sqrt{p} D_p + p/\alpha_p)/\varepsilon)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$.*

Corollary 6.13. *Suppose that $\mathcal{C}$ is (α_p, q_p)-strongly convex with respect to the ℓ_p-norm, where $q_p \in [2, +\infty[$. Then by running FW on problem* (6.1) *with the closed-loop strat-*

egy (6.6)*, we obtain a point* $x \in \mathcal{C}$ *with cardinality* $\mathcal{O}((pD_p^2)^{(q_p-1)/q_p} + p/\alpha_p^{2/q_p})/\varepsilon^{2(q_p-1)/q_p})$ *satisfying* $\|x - x^*\|_p \leqslant \varepsilon$.

Proof. The result follows from Theorem 6.7 and Lemma 6.1, with $\beta_1 \leqslant \sqrt{2}$ and $1/\beta_2 \leqslant 2 + \sqrt{2}$. □

6.5 The case $p \in [1, 2[\cup \{+\infty\}$

In this section, we study the approximate Carathéodory problem when $p \in [1, 2[\cup \{+\infty\}$. In this case, the function $x \in \mathbb{R}^n \mapsto (1/2)\|x - x^*\|_p^2$ is no longer smooth so we cannot apply the Frank-Wolfe algorithm directly. The problem is to find a sparse approximate solution to

$$\min_{x \in \mathcal{C}} \|x - x^*\|_p, \tag{6.7}$$

where the objective $x \in \mathbb{R}^n \mapsto \|x - x^*\|_p$ is convex and 1-Lipschitz continuous with respect to the ℓ_p-norm, by the triangle inequality, but not smooth. Note that, compared to problem (6.1) for $p \in [2, +\infty[$, we removed the square so that the objective is Lipschitz continuous.

We will present two methods to find an ε-approximate solution to problem (6.7) with the same cardinality guarantees. When $p \in [1, 2[$, we ensure a cardinality bound $\mathcal{O}(n^{(2-p)/p} D_2^2/\varepsilon^2)$. Compared to the bound $\mathcal{O}((1/p)^{1/(p-1)}(D_p/\varepsilon)^{p/(p-1)})$ obtained from [13, Lem. D] for $p \in]1, 2[$ (see [62]), it improves the dependence to the accuracy ε but introduces a factor that is linear in the dimension n in the worst case: $n^{(2-p)/p} < n$ for all $p \in]1, 2[$. This is probably due to our approach considering the ℓ_2-norm and assuming the functions to be nondifferentiable, while they are only nonsmooth when $p \in]1, 2[$. When $p = +\infty$, we ensure a cardinality bound $\mathcal{O}(D_2^2/\varepsilon^2)$. This is a dimension-free result compared to the bound $\mathcal{O}(\ln(n) D_\infty^2/\varepsilon^2)$ from [3].

Similarly to the case $p \in [2, +\infty[$, we will present the convergence rate of a variant of

the Frank-Wolfe algorithm, then deduce a bound for the approximate Carathéodory problem. We consider the general problem

$$\min_{x \in \mathcal{C}} f(x), \tag{6.8}$$

where $f \colon \mathbb{R}^n \to \mathbb{R}$ is a convex, continuous, but possibly nonsmooth function (Assumption 6.14). We can smoothen f via its Moreau envelope $f_\beta \colon x \in \mathbb{R}^n \mapsto \min_{y \in \mathbb{R}^n} f(y) + (1/(2\beta))\|x - y\|_2^2$ [102], where $\beta > 0$ is the smoothing parameter. The Moreau envelope is a smooth convex function (Lemma 6.15). The proximity operator of f is $\operatorname{prox}_f \colon x \in \mathbb{R}^n \mapsto \arg\min_{y \in \mathbb{R}^n} f(y) + (1/2)\|x - y\|_2^2$ [101].

Assumption 6.14. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set with diameter D_2 and $f \colon \mathbb{R}^n \to \mathbb{R}$ be a convex G_2-Lipschitz continuous function, all with respect to the ℓ_2-norm.*

Lemma 6.15 ([6, Prop. 12.15 and Prop. 12.30])**.** *Let $f \colon \mathbb{R}^n \to \mathbb{R}$ be a convex continuous function and $\beta > 0$. Then $f_\beta \colon x \in \mathbb{R}^n \mapsto \min_{y \in \mathbb{R}^n} f(y) + (1/(2\beta))\|x - y\|_2^2$ is convex and $1/\beta$-smooth with respect to the ℓ_2-norm. Its gradient at $x \in \mathbb{R}^n$ is*

$$\nabla f_\beta(x) = \frac{1}{\beta}(x - \operatorname{prox}_{\beta f}(x)). \tag{6.9}$$

Our analysis will rely on the ℓ_2-norm. Lemma 6.16 states the properties of the objective in the approximate Carathéodory problem (6.7) with respect to the ℓ_2-norm.

Lemma 6.16. *Let $\mathcal{C} \subset \mathbb{R}^n$ be a compact convex set, $x^* \in \mathcal{C}$, $p \in [1, 2[\cup \{+\infty\}$, and $f \colon x \in \mathbb{R}^n \mapsto \|x - x^*\|_p$. Then f is convex and Lipschitz continuous with respect to the ℓ_2-norm, with constant $n^{1/p-1/2}$ if $p \in [1, 2[$, else 1 if $p = +\infty$.*

Proof. By the triangle inequality, the function f is 1-Lipschitz continuous with respect to the ℓ_p-norm. We conclude using $\|\cdot\|_p \leqslant n^{1/p-1/2}\|\cdot\|_2$ for $p \in [1, 2[$ and $\|\cdot\|_\infty \leqslant \|\cdot\|_2$. □

6.5.1 A nonsmooth variant of the Frank-Wolfe algorithm

The first method to solve problem (6.8) is to use a variant of the Frank-Wolfe algorithm, the Hybrid Conditional Gradient-Smoothing algorithm (HCGS, Algorithm 6.2), developed in [1] for addressing composite convex problems. The idea is to design a strategy $(\beta_t)_{t\in\mathbb{N}}$ and to use the gradient $\nabla f_{\beta_t}(x_t)$ as a surrogate in the Frank-Wolfe algorithm.

Algorithm 6.2 Hybrid Conditional Gradient-Smoothing (HCGS)

Input: Start point $x_0 \in \mathcal{C}$, smoothing strategy $(\beta_t)_{t\in\mathbb{N}} \subset \mathbb{R}_{++}$, step-size strategy $(\gamma_t)_{t\in\mathbb{N}} \subset [0,1]$.

1: **for** $t = 0$ **to** $T-1$ **do**
2: $\quad v_t \leftarrow \arg\min_{v\in\mathcal{C}} \langle v, \nabla f_{\beta_t}(x_t)\rangle$ ▷ see (6.9)
3: $\quad x_{t+1} \leftarrow x_t + \gamma_t(v_t - x_t)$
4: **end for**

Theorem 6.17 presents the convergence rate of HCGS. It is adapted from [1] and uses a slightly different smoothing strategy. Corollaries 6.18–6.19 present the cardinality bounds for the approximate Carathéodory problem using Lemma 6.16.

Theorem 6.17 ([1])**.** *Let Assumption 6.14 hold and consider HCGS (Algorithm 6.2) with the open-loop strategy* (2.4) *and* $\beta_t \leftarrow 2(D_2/G_2)/\sqrt{t+2}$ *for all* $t \in \mathbb{N}$*. Then for all* $t \geqslant 2$*,*

$$f(x_t) - \min_{\mathcal{C}} f \leqslant \frac{4G_2D_2}{\sqrt{t+1}}.$$

Proof. Let $x^* \in \arg\min_{\mathcal{C}} f$ and $t \in \mathbb{N}$. By Lemma 6.15, f_{β_t} is convex and $1/\beta_t$-smooth with respect to the ℓ_2-norm. By optimality of v_t (Line 2), we have

$$\begin{aligned} f_{\beta_t}(x_{t+1}) &\leqslant f_{\beta_t}(x_t) + \gamma_t\langle v_t - x_t, \nabla f_{\beta_t}(x_t)\rangle + \frac{\gamma_t^2}{2\beta_t}\|v_t - x_t\|_2^2 \\ &\leqslant f_{\beta_t}(x_t) + \gamma_t\langle x^* - x_t, \nabla f_{\beta_t}(x_t)\rangle + \frac{\gamma_t^2}{2\beta_t}\|v_t - x_t\|_2^2 \\ &\leqslant f_{\beta_t}(x_t) + \gamma_t(f_{\beta_t}(x^*) - f_{\beta_t}(x_t)) + \frac{\gamma_t^2}{2\beta_t}\|v_t - x_t\|_2^2. \end{aligned}$$

By [1, Lem. 4.2], $f_{\beta_t}(x^*) \leqslant f(x^*)$ and $f_{\beta_{t+1}}(x_{t+1}) \leqslant f_{\beta_t}(x_{t+1}) + (\beta_t - \beta_{t+1})G_2^2/2$, so

$$\begin{aligned} f_{\beta_{t+1}}(x_{t+1}) - f(x^*) &\leqslant (1-\gamma_t)(f_{\beta_t}(x_t) - f(x^*)) + \frac{\gamma_t^2 D_2^2}{2\beta_t} + \frac{(\beta_t - \beta_{t+1})G_2^2}{2} \\ &= \frac{t}{t+2}(f_{\beta_t}(x_t) - f(x^*)) + G_2 D_2 \frac{\sqrt{t+2}}{(t+2)^2} + G_2 D_2 \frac{\sqrt{t+3} - \sqrt{t+2}}{\sqrt{(t+3)(t+2)}}. \end{aligned}$$

Let $\varepsilon_s = f_{\beta_s}(x_s) - f(x^*)$ for all $s \in \mathbb{N}$. Then

$$\begin{aligned} (t+1)(t+2)\varepsilon_{t+1} &= t(t+1)\varepsilon_t + G_2 D_2 \frac{t+1}{\sqrt{t+2}} + G_2 D_2 \frac{(t+1)\sqrt{t+2}}{\sqrt{t+3}}(\sqrt{t+3} - \sqrt{t+2}) \\ &\leqslant t(t+1)\varepsilon_t + G_2 D_2 \sqrt{t+2} + G_2 D_2 \frac{(t+1)\sqrt{t+2}}{\sqrt{t+3}}(\sqrt{t+3} - \sqrt{t+2}). \end{aligned}$$

By telescoping for $s \in [\![0, t]\!]$, we obtain

$$\begin{aligned} (t+1)(t+2)\varepsilon_{t+1} &= G_2 D_2 \sum_{s=0}^{t} \sqrt{s+2} + G_2 D_2 \sum_{s=0}^{t} \frac{(s+1)\sqrt{s+2}}{\sqrt{s+3}}(\sqrt{s+3} - \sqrt{s+2}) \\ &\leqslant G_2 D_2 \sum_{s=0}^{t} \int_s^{s+1} \sqrt{u+2}\,\mathrm{d}u + G_2 D_2 \sum_{s=0}^{t} \frac{(s+1)\sqrt{s+2}}{\sqrt{s+3}}\sqrt{s+3} \\ &\quad - G_2 D_2 \sum_{s=0}^{t} \frac{s\sqrt{s+1}}{\sqrt{s+2}}\sqrt{s+2} \\ &= G_2 D_2 \left[\frac{2}{3}(u+2)^{3/2}\right]_0^{t+1} + G_2 D_2 \sum_{s=0}^{t} \frac{(s+1)\sqrt{s+2}}{\sqrt{s+3}}\sqrt{s+3} \\ &\quad - G_2 D_2 \sum_{s=-1}^{t-1} \frac{(s+1)\sqrt{s+2}}{\sqrt{s+3}}\sqrt{s+3} \\ &\leqslant \frac{2}{3} G_2 D_2 (t+3)^{3/2} + G_2 D_2 (t+1)\sqrt{t+2}. \end{aligned}$$

Thus,

$$\begin{aligned} f_{\beta_{t+1}}(x_{t+1}) - f(x^*) &\leqslant \frac{2}{3} G_2 D_2 \frac{(t+3)^{3/2}}{(t+1)(t+2)} + \frac{G_2 D_2}{\sqrt{t+2}} \\ &\leqslant \frac{3 G_2 D_2}{\sqrt{t+2}}, \end{aligned}$$

where the last inequality holds for all $t \geqslant 1$. By [1, Lem. 4.2], $f(x_{t+1}) \leqslant f_{\beta_{t+1}}(x_{t+1}) + \beta_{t+1}G_2^2/2$. Therefore, for all $t \geqslant 1$,

$$\begin{aligned} f(x_{t+1}) - f(x^*) &\leqslant \frac{3G_2D_2}{\sqrt{t+2}} + \frac{G_2D_2}{\sqrt{t+3}} \\ &\leqslant \frac{4G_2D_2}{\sqrt{t+2}}. \end{aligned}$$

□

Corollary 6.18. *Let $p \in [1, 2[$. By running HCGS (Algorithm 6.2) on problem* (6.7) *with the open-loop strategy* (2.4) *and $\beta_t \leftarrow 2(D_2/n^{1/p-1/2})/\sqrt{t+2}$ for all $t \in \mathbb{N}$, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(n^{(2-p)/p}D_2^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$.*

Corollary 6.19. *By running HCGS (Algorithm 6.2) on problem* (6.7) *with the open-loop strategy* (2.4) *and $\beta_t \leftarrow 2D_2/\sqrt{t+2}$ for all $t \in \mathbb{N}$, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(D_2^2/\varepsilon^2)$ satisfying $\|x - x^*\|_\infty \leqslant \varepsilon$.*

6.5.2 Applying Frank-Wolfe to the smoothed objective

We can view HCGS as applying one iteration of the Frank-Wolfe algorithm to the sequence of problems

$$\min_{x \in \mathcal{C}} f_{\beta_t}(x).$$

However, if the desired level of accuracy $\varepsilon > 0$ is given, which is probably the case in the approximate Carathéodory problem, then we can apply the Frank-Wolfe algorithm to the fixed problem

$$\min_{x \in \mathcal{C}} f_{\beta}(x), \tag{6.10}$$

where $\beta \leftarrow \varepsilon/G_2^2$, to obtain a solution to the original problem (6.8). Theorem 6.20 formalizes this statement, and Corollaries 6.21–6.22 present the cardinality bounds for the approximate Carathéodory problem using Lemma 6.16.

Theorem 6.20. *Let Assumption 6.14 hold. Let $\varepsilon > 0$ be the desired level of accuracy and set $\beta \leftarrow \varepsilon/G_2^2$. Then by running FW (Algorithm 2.1) on problem* (6.10) *with the open-loop strategy* (2.4)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(G_2^2 D_2^2/\varepsilon^2)$ satisfying $f(x) - \min_{\mathcal{C}} f \leqslant \varepsilon$.*

Proof. By Lemma 6.15, f_β is convex and $1/\beta$-smooth with respect to the ℓ_2-norm. By Theorem 2.6, after $\lfloor 4G_2^2 D_2^2/\varepsilon^2 \rfloor$ iterations, we have a point $x \in \mathcal{C}$ such that

$$\begin{aligned} f_\beta(x) - \min_{\mathcal{C}} f_\beta &\leqslant \frac{2D_2^2/\beta}{\lfloor 4G_2^2 D_2^2/\varepsilon^2 \rfloor + 2} \\ &\leqslant \frac{2G_2^2 D_2^2/\varepsilon}{4G_2^2 D_2^2/\varepsilon^2} \\ &= \frac{\varepsilon}{2}. \end{aligned}$$

By [1, Lem. 4.2], $f_\beta \leqslant f \leqslant f_\beta + \beta G_2^2/2$ so

$$\begin{aligned} f(x) - \min_{\mathcal{C}} f &\leqslant f_\beta(x) + \frac{\beta G_2^2}{2} - \min_{\mathcal{C}} f_\beta \\ &\leqslant \varepsilon. \end{aligned}$$

□

Corollary 6.21. *Let $p \in [1, 2[$. Let $\varepsilon > 0$ be the desired level of accuracy and set $\beta \leftarrow \varepsilon/n^{(2-p)/p}$. Then by running FW (Algorithm 2.1) on problem* (6.10) *with the open-loop strategy* (2.4)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(n^{(2-p)/p} D_2^2/\varepsilon^2)$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$.*

Corollary 6.22. *Let $\varepsilon > 0$ be the desired level of accuracy and set $\beta \leftarrow \varepsilon$. Then by running FW (Algorithm 2.1) on problem* (6.10) *with the open-loop strategy* (2.4)*, we obtain a point*

$x \in \mathcal{C}$ *with cardinality* $\mathcal{O}(D_2^2/\varepsilon^2)$ *satisfying* $\|x - x^*\|_\infty \leqslant \varepsilon$.

6.6 Sparse approximate projections in the ℓ_p-norm

The Frank-Wolfe algorithm can also be used to find sparse approximate projections in the ℓ_p-norm, where $p \in [1, +\infty]$. When $p \in [2, +\infty[$, this is problem (6.1) with $x^* \notin \mathcal{C}$. Then the same cardinality bounds from Corollaries 6.9–6.10, 6.12, and 6.13 hold with

$$\|x - x^*\|_p^2 - \operatorname{dist}_p(x^*, \mathcal{C})^2 \leqslant \varepsilon^2. \tag{6.11}$$

Furthermore, Corollaries 6.23–6.24 follow from Theorems 6.4 and 6.6, together with Lemma 6.1. Note that to apply Corollary 6.10 here, D_* is defined with respect to $\operatorname{proj}_p(x^*, \mathcal{C})$ instead of x^*. When $p \in [1, 2[\cup \{+\infty\}$, then the same cardinality bounds from Corollaries 6.18–6.19 and 6.21–6.22 hold with

$$\|x - x^*\|_p - \operatorname{dist}_p(x^*, \mathcal{C}) \leqslant \varepsilon.$$

Corollary 6.23. *Let* $p \in [2, +\infty[$ *and* $x^* \in \mathbb{R}^n \backslash \mathcal{C}$. *Suppose that* $\mathcal{C}$ *is* α_p*-strongly convex with respect to the* ℓ_p*-norm. Then by running FW on problem* (6.1) *with the closed-loop strategy* (6.6)*, we obtain a point* $x \in \mathcal{C}$ *with cardinality* $\mathcal{O}(p/(c_p \alpha_p) \cdot \ln(1/\varepsilon))$ *satisfying*

$$\|x - x^*\|_p^2 - \operatorname{dist}_p(x^*, \mathcal{C})^2 \leqslant \varepsilon^2,$$

where $c_p = \operatorname{dist}_p(x^*, \mathcal{C}) > 0$.

Proof. The bound follows from Theorem 6.4 and Lemma 6.1. Since for $f \colon x \in \mathbb{R}^n \mapsto (1/2)\|x - x^*\|_p^2$, we have $\nabla f(x) = \|x - x^*\|_p^{2-p} \operatorname{sign}(x - x^*)|x - x^*|^{p-1}$ for all $x \in \mathbb{R}^n$,

where $\operatorname{sign}(x-x^*)|x-x^*|^{p-1} = (\operatorname{sign}([x-x^*]_i)|[x-x^*]_i|^{p-1})_{i\in[\![1,n]\!]} \in \mathbb{R}^n$, we obtain

$$\begin{aligned}\|\nabla f(x)\|_{p/(p-1)} &= \|x-x^*\|_p^{2-p} \|\operatorname{sign}(x-x^*)|x-x^*|^{p-1}\|_{p/(p-1)} \\ &= \|x-x^*\|_p^{2-p} \left(\sum_{i=1}^n |[x-x^*]_i|^p\right)^{(p-1)/p} \\ &= \|x-x^*\|_p^{2-p+p-1} \\ &= \|x-x^*\|_p.\end{aligned}$$

Thus, $c_p = \min_{x\in\mathcal{C}} \|\nabla f\|_{p/(p-1)} = \operatorname{dist}_p(x^*,\mathcal{C})$. □

Corollary 6.24. *Let $p \in [2,+\infty[$ and $x^* \in \mathbb{R}^n\backslash\mathcal{C}$. Suppose that $\mathcal{C}$ is (α_p, q_p)-uniformly convex with respect to the ℓ_p-norm, where $q_p > 2$. Then by running FW on problem* (6.1) *with the closed-loop strategy* (6.6)*, we obtain a point $x \in \mathcal{C}$ with cardinality $\mathcal{O}(((pD_p^2)^{(q_p-2)/q_p}(1+\beta_1) + (p/\beta_2)/(\alpha_p c_p)^{2/q})/\varepsilon^{2(q_p-2)/q_p})$ satisfying*

$$\|x-x^*\|_p^2 - \operatorname{dist}_p(x^*,\mathcal{C})^2 \leqslant \varepsilon^2,$$

where $\beta_1 = (2-2^{(q_p-2)/q_p})/(2^{(q_p-2)/q_p}-1)$, $\beta_2 = (q_p-2)/q_p - (2/q_p)(2^{(q_p-2)/q_p}-1)$, and $c_p = \operatorname{dist}_p(x^*,\mathcal{C}) > 0$.

Proof. The proof follows the same arguments as in the proof of Corollary 6.23. □

Remark 6.25. *In* (6.11) *and Corollaries 6.23–6.24, if $p = 2$ then the same results hold with*

$$\|x - \operatorname{proj}_2(x^*,\mathcal{C})\|_p \leqslant \varepsilon.$$

Indeed, let $\pi_2 = \mathrm{proj}_2(x^*, \mathcal{C})$. *Then* $\langle x - \pi_2, x^* - \pi_2 \rangle \leqslant 0$, *so*

$$\begin{aligned}\|x - x^*\|_2^2 &= \|x - \pi_2\|_2^2 + \|\pi_2 - x^*\|_2^2 + 2\langle x - \pi_2, \pi_2 - x^* \rangle \\ &\geqslant \|x - \pi_2\|_2^2 + \|\pi_2 - x^*\|_2^2.\end{aligned}$$

6.7 Computational experiments

We compare the cardinality bounds of the solutions obtained by FW (Algorithm 2.1), FCFW (Algorithm 2.2), NEP-FW (Algorithm 6.1), and the Away-Step Frank-Wolfe algorithm (AFW) [131] on the approximate Carathéodory problem (6.1) for $p \in [2, +\infty[$. While FW moves only towards vertices, AFW is a variant of FW that allows to move also away from vertices, and converges at a linear rate for smooth strongly convex objectives [78]. Although the objective in (6.1) is not strongly convex with respect to the ℓ_p-norm when $p \in]2, +\infty[$, it is still interesting to investigate its performance.

The experiments were run on a laptop under Linux Ubuntu 20.04 with Intel Core i7-10750H. The code is available at https://github.com/cyrillewcombettes/approxcara.

6.7.1 Dense vs. sparse target x^*

We generated a set $\mathcal{V} \subset \mathbb{R}^{500}$ of 501 random points and let $\mathcal{C} = \mathrm{conv}\, \mathcal{V}$. Then, we generated the target point $x^* \in \mathcal{C}$ as a random convex combination of points in $\mathcal{V}$ or as a sparse random convex combination of points in $\mathcal{V}$, i.e., we randomly selected 25 points from $\mathcal{V}$ and we generated x^* as a convex combination of these points only. Figure 6.1 plots the distance of the iterate x_t to the target x^* in the ℓ_p-norm as a function of its cardinality, as given by the construction of the algorithm, for three arbitrary values $p = 2$, $p = 3$, and $p = 7$.

As expected, FCFW is the best performing algorithm, followed by AFW. In the instance "Sparse x^*, $p = 2$", we can see that AFW took a full away step, which decreased the cardinality of the iterate by 1. In the limit, NEP-FW improves on FW particularly when the

target x^* is sparse; note that we chose $x_0 \in \mathcal{V}$ randomly and did not use a warm start. Only FCFW is able to recover an optimal convex decomposition on each instance, i.e., to find a solution with arbitrary accuracy and cardinality no greater than $\mathrm{card}(x^*)$. Table 6.3 reports the cardinality of the solution obtained by each algorithm to reach accuracy $\varepsilon = 0.02$.

Table 6.3: Cardinality of the first iterate x_t satisfying $\|x_t - x^*\|_p < 0.02$.

$\mathrm{card}(x^*)$	p	**FW**	**NEP-FW**	**AFW**	**FCFW**
501	2	470	496	453	368
25	2	43	25	26	25
501	3	417	413	388	275
25	3	42	29	25	22
501	7	349	325	294	175
25	7	49	40	28	22

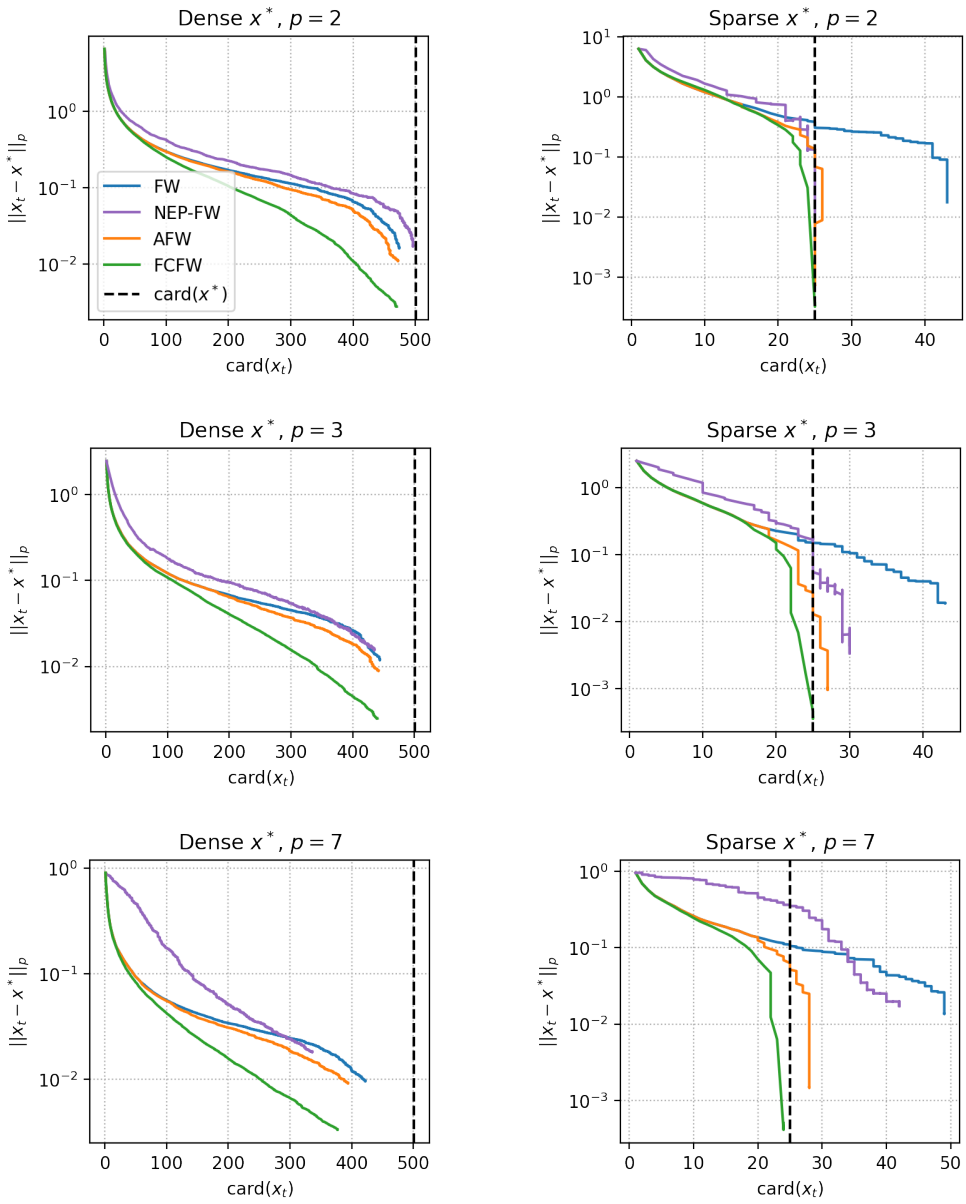

Figure 6.1: Accuracy vs. cardinality of the iterates generated by FW, NEP-FW, AFW, and FCFW.

6.7.2 Lower bound

We compare FW, AFW, and FCFW to the lower bound in [100, Sec. 5.1]. Let $\mathcal{C}$ be the convex hull of the ℓ_p-normalized columns of the Hadamard matrix H_n of dimension n, i.e., $\mathcal{C} = \operatorname{conv}(H_n/n^{1/p})$, and let $x^* = (H_n/n^{1/p})1/n = e_1/n^{1/p}$ be the uniform convex combination of the columns, where $e_1 \in \mathbb{R}^n$ denotes the first canonical vector. In this setting, [100, Thm. 5.3] claims that for any $x \in \operatorname{conv}(H_n/n^{1/p})$ satisfying $\|x - x^*\|_p \leqslant \varepsilon$,

then x has cardinality $s \geqslant \min\{1/\varepsilon^2, n\}$. However, in their proof they use the inequality

$$\frac{1}{\varepsilon^2 + 1/n} \geqslant \frac{1}{\max\{\varepsilon^2, 1/n\}},$$

which does not hold. Hence, for completeness, we present a minor correction to their lower bound in Theorem 6.26.

Theorem 6.26. *Let $p \in [2, +\infty[$, $n \in \{2^k \mid k \in \mathbb{N}\}$, H_n be the Hadamard matrix of dimension n, $\mathcal{C} = \operatorname{conv}(H_n/n^{1/p})$ be the convex hull of the ℓ_p-normalized columns of H_n, and $x^* = e_1/n^{1/p} \in \mathcal{C}$. Let $\varepsilon > 0$ and $x \in \mathcal{C}$ such that $\|x - x^*\|_p \leqslant \varepsilon$. Then x is the convex combination of at least $1/(\varepsilon^2 + 1/n)$ vertices.*

Figure 6.2 compares FW, AFW, FCFW, and the corrected lower bound $s \in [\![1, n]\!] \mapsto \varepsilon = \sqrt{1/s - 1/n}$, for $n = 64$ and two arbitrary values $p = 4$ and $p = 13$.

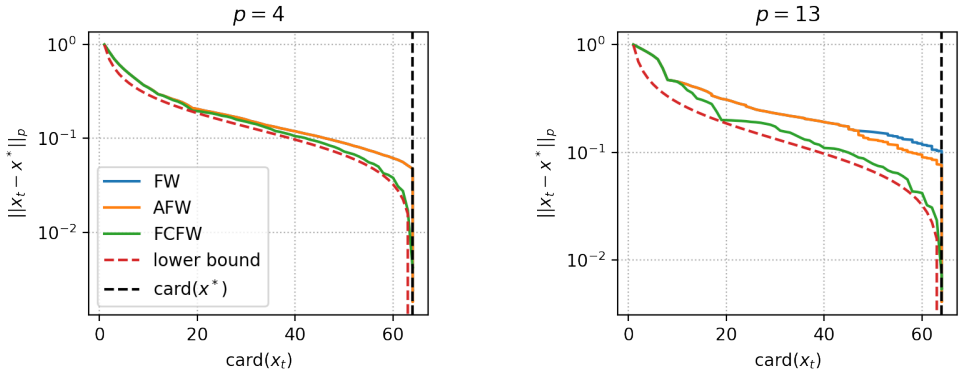

Figure 6.2: Cardinality of the iterates produced by FW, AFW, and FCFW, and the lower bound from Theorem 6.26.

FCFW again demonstrates its performance and almost matches the lower bound. This highlights its significance for the approximate Carathéodory problem. However, it remains an open problem to derive a precise convergence rate for FCFW, as the current analysis is transferred from the analyses of FW and AFW [78].

6.8 Final remarks

We have demonstrated that the Frank-Wolfe algorithm provides a simple implementation of a solution with cardinality $\mathcal{O}(pD_p^2/\varepsilon^2)$ to the approximate Carathéodory problem in the ℓ_p-norm, where $p \in [2, +\infty[$. When x^* is in the (relative) interior of $\mathcal{C}$, which may be likely in practice, the algorithm naturally adapts and generates a solution with cardinality $\mathcal{O}(p(D_p/r_p)^2 \ln(1/\varepsilon))$, where r_p is the radius of an (affine) ball centered at x^* and contained in $\mathcal{C}$. This is in contrast with the method in [100] which requires knowledge of r_p. The Frank-Wolfe algorithm also adapts to the geometry of $\mathcal{C}$, and generates a solution with an improved cardinality bound when $\mathcal{C}$ is uniformly convex. When x^* is the convex combination of a subset of vertices with small diameter, better cardinality bounds are obtained via a variant of the Frank-Wolfe algorithm with an enhanced oracle. When $p \in [1, 2[\cup \{+\infty\}$, new bounds are proposed via a nonsmooth variant of the algorithm. Lastly, we addressed the problem of finding sparse approximate projections in the ℓ_p-norm. In practice, when $p \in [2, +\infty[$, the Fully-Corrective Frank-Wolfe algorithm is very efficient and can generate a solution with near-optimal cardinality. However, a precise estimation has yet to be derived.

CHAPTER 7

BLENDED MATCHING PURSUIT: SPARSE OPTIMIZATION OVER THE LINEAR SPAN

We consider an extension of the Frank-Wolfe method to the problem of minimizing a smooth convex function over the linear span of a given set. The extension resembles matching pursuit algorithms, an important class of algorithms in signal processing and machine learning. We present an algorithm combining Frank-Wolfe-like steps with stronger gradient descent steps, and derive sublinear to linear convergence rates according to the smoothness and sharpness orders of the function. In particular, we derive linear rates for a large class of non-strongly convex functions, and we demonstrate in experiments that our algorithm enjoys very fast rates of convergence and wall-clock speed while maintaining a sparsity of iterates very comparable to that of the (much slower) orthogonal matching pursuit.

Based on [22].

7.1 Optimization over the linear span

Let $\mathcal{H}$ be a separable real Hilbert space, $\mathcal{D} \subset \mathcal{H}$ be a dictionary, and $f\colon \mathcal{H} \to \mathbb{R}$ be a smooth convex function. In this chapter, we aim at finding a sparse (with respect to $\mathcal{D}$) solution to

$$\min_{x \in \mathcal{H}} f(x). \tag{7.1}$$

Together with fast convergence, achieving high sparsity, i.e., keeping the iterates as linear combinations of a *small* number of atoms in the dictionary $\mathcal{D}$, is a primary objective and leads to better generalization, interpretability, and decision-making in machine learning. In signal processing, problem (7.1) encompasses a wide range of applications, including

compressed sensing, signal denoising, and information retrieval, and is often solved with the Matching Pursuit algorithm [96]. Our approach is inspired by the Blended Conditional Gradient algorithm [14], which solves the constrained setting of problem (7.1), i.e., minimizing f over the convex hull $\operatorname{conv} \mathcal{D}$ of the dictionary, and is ultimately based on the Frank-Wolfe algorithm. As introduced in [15], [14] enhanced the vanilla Frank-Wolfe algorithm by replacing the linear minimization oracle with a *weak-separation oracle* and by blending the traditional Frank-Wolfe steps with *lazified* Frank-Wolfe steps and projected gradient steps, while still avoiding projections.

An analogy between Frank-Wolfe algorithms and the unconstrained problem (7.1) was proposed in [88]. They unified the Frank-Wolfe and Matching Pursuit algorithms, and proposed a Generalized Matching Pursuit algorithm (GMP) and a Generalized Orthogonal Matching Pursuit algorithm (OMP) for solving problem (7.1). Essentially, [88] established that GMP corresponds to the vanilla Frank-Wolfe algorithm and OMP corresponds to the Fully-Corrective Frank-Wolfe algorithm. GMP and OMP converge with similar rates in the various regimes, namely with a sublinear rate for smooth convex functions and with a linear rate for smooth strongly convex functions, however they have different advantages: GMP converges (much) faster in wall-clock time while OMP offers (much) sparser iterates. The interest in these algorithms stems from the fact that they work in the general setting of smooth convex functions in Hilbert spaces and that their convergence analyses do not require incoherence or restricted isometry properties (RIP, [16]) of the dictionary, which are quite strong assumptions from an optimization standpoint. In a follow-up work, [89] presented an Accelerated Matching Pursuit algorithm, which we compare our approach to as well.

We aim at unifying the best of GMP (speed) and OMP (sparsity) into a single algorithm by blending them strategically. However, while the overall idea is reasonably natural, we face considerable challenges as many important features of Frank-Wolfe methods do not apply anymore in the Matching Pursuit setting and cannot be as easily overcome as in

[88], requiring a different analysis. For example, Frank-Wolfe duality gaps are not readily available but they are crucial in monitoring the blending, and further key components, such as the weak-separation oracle, require modifications.

Contributions. We propose the *Blended Matching Pursuit* algorithm (BMP), a fast and sparse first-order method for solving problem (7.1). Our method unifies the best of GMP (speed) and OMP (sparsity) into one algorithm, which is of fundamental interest for practitioners. We establish a continuous range of convergence rates between $\mathcal{O}(1/\varepsilon^p)$ and $\mathcal{O}(\ln(1/\varepsilon))$, where $\varepsilon > 0$ is the desired accuracy and $p > 0$ depends on the properties of the function. In particular, we derive linear rates of convergence for a large class of smooth convex but non-strongly convex functions. Lastly, we demonstrate the computational superiority of BMP over state-of-the-art methods, with BMP converging the fastest in wall-clock time while maintaining its iterates at close-to-optimal sparsity, and this without requiring sparsity-inducing constraints.

Outline. We introduce notions and notation in Section 7.2. We present the Blended Matching Pursuit algorithm in Section 7.3 with the convergence analysis in Section 7.3.1. Computational experiments are provided in Section 7.4. Additional experiments and results can be found in Appendix B.4.

7.2 Preliminaries

We work in a separable real Hilbert space $(\mathcal{H}, \langle\cdot,\cdot\rangle)$ with induced norm $\|\cdot\|$. A set $\mathcal{D} \subset \mathcal{H}$ of normalized vectors is a dictionary if it is at most countable and $\operatorname{cl}\operatorname{span}\mathcal{D} = \mathcal{H}$, and in this case its elements are referred to as atoms. For any set $\mathcal{S} \subset \mathcal{H}$, let $\mathcal{S}' = \mathcal{S} \cup -\mathcal{S}$ denote the symmetrization of $\mathcal{S}$ and $D_{\mathcal{S}} = \sup_{u,v\in\mathcal{S}} \|u - v\|$ denote the diameter of $\mathcal{S}$. For problem (7.1) to be feasible, we will assume f to be coercive, i.e., $\lim_{\|x\|\to+\infty} f(x) = +\infty$. Since f is convex, this is actually a mild assumption when $\arg\min_{\mathcal{H}} f \neq \varnothing$.

Let $f\colon \mathcal{H} \to \mathbb{R}$ be a Fréchet differentiable function. In the following, we use extended notions of smoothness and strong convexity by introducing orders, and we weaken and generalize the notion of strong convexity to that of *sharpness*; see, e.g., [120] and [68] for recent work. We say that f is:

(i) *smooth of order* $\ell > 1$ if there exists $L > 0$ such that for all $x, y \in \mathcal{H}$,

$$f(y) - f(x) - \langle y - x, \nabla f(x) \rangle \leqslant \frac{L}{\ell} \|y - x\|^{\ell},$$

(ii) *strongly convex of order* $s > 1$ if there exists $S > 0$ such that for all $x, y \in \mathcal{H}$,

$$f(y) - f(x) - \langle y - x, \nabla f(x) \rangle \geqslant \frac{S}{s} \|y - x\|^{s},$$

(iii) *sharp of order* $\theta \in]0, 1[$ *on* $\mathcal{K}$ if $\mathcal{K} \subset \mathcal{H}$ is a bounded set, $\varnothing \neq \arg\min_{\mathcal{H}} f \subset \operatorname{int} \mathcal{K}$, and there exists $\sigma > 0$ such that for all $x \in \mathcal{K}$,

$$\operatorname{dist}\left(x, \arg\min_{\mathcal{H}} f\right) \leqslant \sigma \left(f(x) - \min_{\mathcal{H}} f\right)^{\theta}.$$

The following fact, whose result was already used in [107], provides a bound on the sharpness order of a smooth function.

Fact 7.1. *Let* $f\colon \mathcal{H} \to \mathbb{R}$ *be smooth of order* $\ell > 1$*, convex, and sharp of order* $\theta \in]0, 1[$ *on* $\mathcal{K}$*. Then* $\theta \in]0, 1/\ell]$.

Proof. Let $x \in \mathcal{K} \backslash \arg\min_{\mathcal{H}} f$ and $x^* = \operatorname{proj}(x, \arg\min_{\mathcal{H}} f)$. By sharpness, smoothness, and since $\nabla f(x^*) = 0$,

$$\operatorname{dist}\left(x, \arg\min_{\mathcal{H}} f\right) = \|x - x^*\| \leqslant \sigma (f(x) - f(x^*))^{\theta} \leqslant \sigma \left(\frac{L}{\ell}\right)^{\theta} \|x - x^*\|^{\ell\theta}.$$

Therefore,

$$\frac{1}{\sigma}\left(\frac{\ell}{L}\right)^{\theta} \leqslant \|x - x^*\|^{\ell\theta - 1}.$$

As the left-hand side is constant and x can be arbitrarily close to x^*, we conclude that $\ell\theta \leqslant 1$. □

Fact 7.2. *Let $f\colon \mathcal{H} \to \mathbb{R}$ be a coercive function and $(x_t)_{t\in\mathbb{N}}$ be a sequence of iterates in $\mathcal{H}$ such that $f(x_{t+1}) \leqslant f(x_t)$ for all $t \in \mathbb{N}$. Then $(x_t)_{t\in\mathbb{N}}$ is bounded.*

Proof. By assumption, $f(x_t) \leqslant f(x_0)$ for all $t \in \mathbb{N}$, so $\limsup_{t\to+\infty} f(x_t) \leqslant f(x_0) < +\infty$. Suppose $(x_t)_{t\in\mathbb{N}}$ is unbounded. Then there exists $\varphi\colon \mathbb{N} \to \mathbb{N}$ strictly increasing such that the subsequence $(x_{\varphi(t)})_{t\in\mathbb{N}}$ satisfies $\lim_{t\to+\infty} \|x_{\varphi(t)}\| = +\infty$. By coercivity, this implies that $\lim_{t\to+\infty} f(x_{\varphi(t)}) = +\infty$, and therefore $\limsup_{t\to+\infty} f(x_t) \geqslant +\infty$. This is absurd. □

Lemma 7.3. *Let $f\colon \mathcal{H} \to \mathbb{R}$ be differentiable, $M > 0$, $\mu > 1$, and $x, v \in \mathcal{H}$ such that $\langle v, \nabla f(x)\rangle \leqslant 0$. Define*

$$g : \gamma \in \mathbb{R}_+ \mapsto f(x) + \gamma\langle \nabla f(x), v\rangle + \frac{M}{\mu}\gamma^{\mu}\|v\|^{\mu}.$$

Then

$$\min_{\mathbb{R}_+} g = f(x) - \frac{\langle v, -\nabla f(x)\rangle^{\overline{\mu}}}{\overline{\mu} M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}$$

where $\overline{\mu} = \mu/(\mu-1) > 1$.

Proof. Let $\overline{\mu} = \mu/(\mu-1)$. We have $\overline{\mu} - 1 = 1/(\mu-1)$, and g is differentiable with

$$\begin{aligned}\forall \gamma \in \mathbb{R}_+,\ g'(\gamma) \geqslant 0 &\Leftrightarrow \langle v, \nabla f(x)\rangle + M\gamma^{\mu-1}\|v\|^{\mu} \geqslant 0 \\ &\Leftrightarrow \gamma \geqslant \left(\frac{\langle v, -\nabla f(x)\rangle}{M\|v\|^{\mu}}\right)^{1/(\mu-1)} = \frac{\langle v, -\nabla f(x)\rangle^{\overline{\mu}-1}}{M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}.\end{aligned}$$

Therefore, using $\mu(\overline{\mu}-1)=\overline{\mu}$ and $1-1/\mu=1/\overline{\mu}$,

$$\begin{aligned}\min_{\mathbb{R}_+} g &= f(x)+\frac{\langle v,-\nabla f(x)\rangle^{\overline{\mu}-1}}{M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}\nabla f(x)v+\frac{M}{\mu}\left(\frac{\langle v,-\nabla f(x)\rangle^{\overline{\mu}-1}}{M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}\right)^{\mu}\|v\|^{\mu}\\ &= f(x)-\frac{\langle v,-\nabla f(x)\rangle^{\overline{\mu}}}{M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}+\frac{1}{\mu}\frac{\langle v,-\nabla f(x)\rangle^{\mu(\overline{\mu}-1)}}{M^{\mu(\overline{\mu}-1)-1}\|v\|^{\mu(\overline{\mu}-1)}}\\ &= f(x)-\left(1-\frac{1}{\mu}\right)\frac{\langle v,-\nabla f(x)\rangle^{\overline{\mu}}}{M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}\\ &= f(x)-\frac{\langle v,-\nabla f(x)\rangle^{\overline{\mu}}}{\overline{\mu}M^{\overline{\mu}-1}\|v\|^{\overline{\mu}}}.\end{aligned}$$

□

7.2.1 On sharpness and strong convexity

Notice that if $f\colon \mathcal{H}\to\mathbb{R}$ is Fréchet differentiable and strongly convex of order $s>1$, then $\operatorname{card}(\arg\min_{\mathcal{H}} f)=1$. Let $\{x^*\}=\arg\min_{\mathcal{H}} f$. It follows directly from $\nabla f(x^*)=0$ that for any bounded set $\mathcal{K}\subset\mathcal{H}$ such that $x^*\in\operatorname{int}\mathcal{K}$, f is sharp of order $\theta=1/s$ on $\mathcal{K}$. Thus, strong convexity implies sharpness. However, not every sharp function is strongly convex; moreover, the next example shows that not every sharp and convex function is strongly convex.

Example 7.4 (Distance to a convex set)**.** *Let $\mathcal{C}\subset\mathcal{H}$ be a nonempty, closed, and bounded convex set, and $\mathcal{K}\subset\mathcal{H}$ be a bounded set such that $\mathcal{C}\subset\operatorname{int}\mathcal{K}$. The function $f\colon x\in\mathcal{H}\mapsto\operatorname{dist}(x,\mathcal{C})^2=\|x-\operatorname{proj}(x,\mathcal{C})\|^2$ is convex, and it is sharp of order $\theta=1/2$ on $\mathcal{K}$. Indeed, since $\arg\min_{\mathcal{H}} f=\mathcal{C}$ and $\min_{\mathcal{H}} f=0$, we have for all $x\in\mathcal{K}$,*

$$\operatorname{dist}\left(x,\arg\min_{\mathcal{H}} f\right)=\|x-\operatorname{proj}(x,\mathcal{C})\|=\left(f(x)-\min_{\mathcal{H}} f\right)^{1/2}.$$

Now, suppose $\mathcal{C}$ contains more than one element. Then, f has more than one minimizer. However, a function that is strongly convex of order $s>1$ has no more than one minimizer. Therefore, f cannot be strongly convex of order s, for all $s>1$. Notice that f is also a

smooth function, of order $\ell = 2$.

Hence, sharpness is a more general notion of strong convexity. It is a local condition around the optimal solutions while strong convexity is a global condition. In fact, building on the Łojasiewicz inequality of [91], [11, Eq. (15)] showed that sharpness always holds in finite dimensional spaces for reasonably well-behaved convex functions; see Lemma 7.5. Polynomial convex functions, the ℓ_p-norms, the Huber loss (see Appendix B.4.4), and the rectifier ReLU are simple examples of such functions.

Lemma 7.5. *Let* $f : \mathbb{R}^n \to]-\infty, +\infty]$ *be a lower semicontinuous, convex, and subanalytic function with* $\{x \in \mathbb{R}^n \mid 0 \in \partial f(x)\} \neq \varnothing$. *Then for any bounded set* $\mathcal{K} \subset \mathbb{R}^n$, *there exists* $\theta \in]0, 1[$ *and* $\sigma > 0$ *such that for all* $x \in \mathcal{K}$,

$$\operatorname{dist}\left(x, \operatorname*{arg\,min}_{\mathbb{R}^n} f\right) \leqslant \sigma \left(f(x) - \min_{\mathbb{R}^n} f\right)^{\theta}.$$

Strong convexity is a standard requirement to prove linear convergence rates on smooth convex objectives but, regrettably, this considerably restricts the set of candidate functions. For our Blended Matching Pursuit algorithm, we will only require sharpness to establish linear convergence rates, thus including a larger class of functions.

7.2.2 Matching Pursuit algorithms

For $y \in \mathcal{H}$ and $f : x \in \mathcal{H} \mapsto \|y - x\|^2/2$, problem (7.1) falls in the area of sparse recovery and is often solved via the Matching Pursuit algorithm [96]. The algorithm recovers a sparse representation of the signal y from the dictionary $\mathcal{D}$ by sequentially *pursuing* the best *matching* atom. At each iteration, it searches for an atom $v_t \in \mathcal{D}$ most correlated with the residual $y - x_t$, i.e., $v_t \in \arg\max_{v \in \mathcal{D}} |\langle v, y - x_t \rangle|$, and adds it to the linear decomposition of the current iterate x_t to form the new iterate x_{t+1}, keeping track of the *active set* $\mathcal{S}_{t+1} = \mathcal{S}_t \cup \{v_t\}$. However, this does not prevent the algorithm from selecting atoms that have already been added in earlier iterations or that are redundant, hence affecting sparsity. The

Orthogonal Matching Pursuit variant [110, 33] overcomes this by computing the new iterate as the projection of the signal y onto $\mathcal{S}_t \cup \{v_t\}$. Thus, $y - x_{t+1}$ becomes orthogonal to the active set.

In order to solve problem (7.1) for any smooth convex objective, [88] proposed the Generalized Matching Pursuit (GMP) and Generalized Orthogonal Matching Pursuit algorithms (Algorithm 7.1); slightly abusing notation we will refer to the latter simply as Orthogonal Matching Pursuit (OMP). The atom selection subroutine is implemented with a Frank-Wolfe linear minimization oracle $\arg\min_{v \in \mathcal{D}'} \langle v, \nabla f(x_t) \rangle$ (Line 3). The solution $v_t \in \mathcal{D}'$ to this oracle is guaranteed to be a descent direction as it satisfies $\langle v_t, \nabla f(x_t) \rangle \leqslant 0$ by symmetry of $\mathcal{D}'$, and $\langle v_t, \nabla f(x_t) \rangle = 0$ if and only if $x_t \in \arg\min_{\mathcal{H}} f$. Notice that for $y \in \mathcal{H}$ and $f \colon x \in \mathcal{H} \mapsto \|y - x\|^2/2$, the GMP and OMP variants of Algorithm 7.1 recover the original Matching Pursuit and Orthogonal Matching Pursuit algorithms respectively. In particular, up to a sign which does not affect the sequence of iterates, $\arg\max_{v \in \mathcal{D}} |\langle v, y - x_t \rangle| \Leftrightarrow \arg\min_{v \in \mathcal{D}'} \langle v, \nabla f(x_t) \rangle$. In practice, the main difference in the case of general smooth convex functions is that the OMP variant (Line 6) is much more expensive, as a closed-form solution to this projection step is not available anymore. Hence, Line 6 is typically a sequence of projected (onto $\operatorname{span} \mathcal{S}_{t+1}$) gradient steps and OMP is significantly slower than GMP to converge.

Algorithm 7.1 Generalized/Orthogonal Matching Pursuit (GMP/OMP)

Input: Start atom $x_0 \in \mathcal{D}$.

1: $\mathcal{S}_0 \leftarrow \{x_0\}$
2: **for** $t = 0$ **to** $T - 1$ **do**
3: $\quad v_t \leftarrow \arg\min_{v \in \mathcal{D}'} \langle v, \nabla f(x_t) \rangle$
4: $\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
5: $\quad$ *GMP variant*: $x_{t+1} \leftarrow \arg\min_{x \in x_t + \mathbb{R} v_t} f(x)$
6: $\quad$ *OMP variant*: $x_{t+1} \leftarrow \arg\min_{x \in \operatorname{span} \mathcal{S}_{t+1}} f(x)$
7: **end for**

7.2.3 Weak-separation oracle

We present in Oracle 7.2 the weak-separation oracle, a *modified* version of the one first introduced in [15] and used in, e.g., [81, 14]. Note that the modification asks for an unconstrained improvement, whereas the original weak-separation oracle required an improvement relative to a reference point. As such, our variant here is even simpler than the original weak-separation oracle. The oracle is called in Line 11 by the Blended Matching Pursuit algorithm.

Oracle 7.2 Weak-separation $\mathrm{LPsep}_{\mathcal{D}}(c, \phi, \kappa)$

Input: Linear objective $c \in \mathcal{H}$, objective value $\phi \leqslant 0$, accuracy $\kappa \geqslant 1$.
Output: Either atom $v \in \mathcal{D}$ such that $\langle v, c \rangle \leqslant \phi/\kappa$ (positive call), or **false** ensuring $\langle z, c \rangle \geqslant \phi$ for all $z \in \operatorname{conv} \mathcal{D}$ (negative call).

The weak-separation oracle determines whether there exists an atom $v \in \mathcal{D}$ such that $\langle v, c \rangle \leqslant \phi/\kappa$, and thereby relaxes the Frank-Wolfe linear minimization oracle. If not, then this implies that $\operatorname{conv} \mathcal{D}$ can be *separated* from the ambient space by c and ϕ with the linear inequality $\langle z, c \rangle \geqslant \phi$ for all $z \in \operatorname{conv} \mathcal{D}$. In practice, the oracle can be efficiently implemented using *caching*, i.e., first testing atoms that were already returned during previous calls as they may satisfy the condition here again. In this case, caching also preserves sparsity. If no active atom satisfies the condition, the oracle can be solved, e.g., by means of a call to a linear optimization oracle; see [15] for an in-depth discussion. Lastly, we would like to briefly note that the parameter κ can be used to further promote positive calls over negative calls, by weakening the improvement requirement and therefore speeding up the oracle. Indeed, only negative calls need a full scan of the dictionary.

7.3 The Blended Matching Pursuit algorithm

We now present our Blended Matching Pursuit algorithm (BMP) in Algorithm 7.3. Note that although we blend steps, we maintain the explicit decomposition of the iterates as linear combinations of the atoms.

Algorithm 7.3 Blended Matching Pursuit (BMP)

Input: Start atom $x_0 \in \mathcal{D}$, accuracy parameters $\kappa \geqslant 1$ and $\eta > 0$, scaling parameter $\tau > 1$.

1: $\mathcal{S}_0 \leftarrow \{x_0\}$
2: $\phi_0 \leftarrow \min_{v \in \mathcal{D}'} \langle v, \nabla f(x_0) \rangle / \tau$
3: **for** $t = 0$ **to** $T - 1$ **do**
4: $\quad v_t^{\text{FW-}\mathcal{S}} \leftarrow \arg\min_{v \in \mathcal{S}_t'} \langle v, \nabla f(x_t) \rangle$
5: $\quad$ **if** $\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t) \rangle \leqslant \phi_t / \eta$ **then**
6: $\quad\quad \tilde{\nabla} f(x_t) \leftarrow \operatorname{proj}(\nabla f(x_t), \operatorname{span} \mathcal{S}_t)$
7: $\quad\quad x_{t+1} \leftarrow \arg\min_{x \in x_t + \mathbb{R}\tilde{\nabla} f(x_t)} f(x)$ ▷ constrained step
8: $\quad\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t$
9: $\quad\quad \phi_{t+1} \leftarrow \phi_t$
10: $\quad$ **else**
11: $\quad\quad v_t \leftarrow \text{LPsep}_{\mathcal{D}'}(\nabla f(x_t), \phi_t, \kappa)$
12: $\quad\quad$ **if** $v_t =$ **false then**
13: $\quad\quad\quad x_{t+1} \leftarrow x_t$ ▷ dual step
14: $\quad\quad\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t$
15: $\quad\quad\quad \phi_{t+1} \leftarrow \phi_t / \tau$
16: $\quad\quad$ **else**
17: $\quad\quad\quad x_{t+1} \leftarrow \arg\min_{x \in x_t + \mathbb{R}v_t} f(x)$ ▷ full step
18: $\quad\quad\quad \mathcal{S}_{t+1} \leftarrow \mathcal{S}_t \cup \{v_t\}$
19: $\quad\quad\quad \phi_{t+1} \leftarrow \phi_t$
20: $\quad\quad$ **end if**
21: $\quad$ **end if**
22: $\quad$ *Optional:* Correct $\mathcal{S}_{t+1}$
23: **end for**

Remark 7.6 (Algorithm design)**.** *BMP actually does not require the atoms to have exactly the same norm and only needs the dictionary to be bounded, whether it be for ensuring the convergence rates or for computations; one could further take advantage of this to add weights to certain atoms. Line 6 is simply taking the component of* $\nabla f(x_t)$ *parallel to* $\operatorname{span} \mathcal{S}_t$*, which can be achieved by basic linear algebra and costs* $\mathcal{O}(n \operatorname{card}(\mathcal{S}_t)^2)$ *when* $\mathcal{H} = \mathbb{R}^n$*. The line searches Lines 7 and 17 can be replaced with explicit step sizes using the smoothness of* f *(Lemma 7.3). The purpose of (the optional) Line 22 is to reoptimize the active set* $\mathcal{S}_{t+1}$*, e.g., by reducing it to a subset that forms a basis for its linear span. One could also obtain further sparsity by removing atoms whose coefficient in the decomposition of*

the iterate is smaller than some threshold $\delta > 0$.

Blending. BMP aims at unifying the speed of GMP and the sparsity of OMP. As seen in Section 7.2.2, an OMP iteration is typically a sequence of projected gradient (PG) steps. The idea is that the sequence of PG steps constituting an OMP iteration is actually overkill: there is a sweet spot where further optimizing over the active space $\operatorname{span} \mathcal{S}_t$ is less effective than adding a new atom and taking a GMP step into a (possibly) new space. However, PG steps have the benefit of preserving sparsity, since no new atom is added. Furthermore, GMP steps require an expensive scan of the dictionary to output the descent direction $v_t^{\text{FW}} \leftarrow \arg\min_{v \in \mathcal{D}'} \langle v, \nabla f(x_t) \rangle$. To remedy this, BMP blends *constrained steps* (PG steps, Line 7) with *full steps* (lazified GMP steps, Line 17) by promoting constrained steps as long as the progress in function value is *comparable* to that of a GMP step, else by taking a full step in an approximate direction v_t (with cheap computation via Oracle 7.2) such that the progress is *comparable* to that of a GMP step. Therefore, to monitor this blending of steps, we wish to compare $\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t) \rangle$ and $\langle v_t, \nabla f(x_t) \rangle$ to $\langle v_t^{\text{FW}}, \nabla f(x_t) \rangle$, which quantities measure the progress in function value offered by a constrained step, a full step, and a GMP step respectively.

Duality gap estimates. The aforementioned comparisons however cannot be made directly as the quantity $\langle v_t^{\text{FW}}, \nabla f(x_t) \rangle$ is (deliberately) not computed; computing it requires an expensive complete scan of the dictionary. Instead, we use an estimation of this quantity, by introducing the *duality gap estimate* $|\phi_t|$. This designation comes from the fact that $-\langle v_t^{\text{FW}}, \nabla f(x_t) \rangle$ is our equivalent of the Frank-Wolfe duality gap (Definition 2.1), and this will guide how we build our estimation. Indeed, since $\mathcal{D}'$ is symmetric and assuming $0 \in \text{intconv}\mathcal{D}'$, there exists (an unknown) $\rho > 0$ such that $\{x_0, \dots, x_T\} \cup \arg\min_{\mathcal{H}} f \subset$

$\rho\,\mathrm{conv}\mathcal{D}'$. Then for all $x^* \in \arg\min_{\mathcal{H}} f$,

$$\begin{aligned}\varepsilon_t = f(x_t) - f(x^*) &\leqslant \langle x_t - x^*, \nabla f(x_t)\rangle \\ &\leqslant \max_{u,v \in \rho\,\mathrm{conv}\mathcal{D}'} \langle u - v, \nabla f(x_t)\rangle \\ &= -2\rho\langle v_t^{\mathrm{FW}}, \nabla f(x_t)\rangle, \end{aligned} \tag{7.2}$$

which is our desired inequality. We set $\phi_0 \leftarrow \langle v_0^{\mathrm{FW}}, \nabla f(x_0)\rangle/\tau$ (Line 2) so $\varepsilon_0 \leqslant 2\tau\rho|\phi_0|$ by (7.2). The criterion in Line 5 compares $\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t)\rangle$ to ϕ_t. If this quantity is below the threshold ϕ_t, then a constrained step is not taken and the weak-separation oracle (Line 11, Oracle 7.2) is called to search for an atom v_t satisfying $\langle v_t, \nabla f(x_t)\rangle \leqslant \phi_t$. If the oracle cannot find such an atom, then a full step is not taken and it returns a *negative call* with the certificate $\langle v_t^{\mathrm{FW}}, \nabla f(x_t)\rangle > \phi_t$. In this case, BMP has detected an improved duality gap estimate and takes a *dual step* (Line 13): by (7.2), this implies that $\varepsilon_t \leqslant 2\rho|\phi_t|$ so with $\phi_{t+1} \leftarrow \phi_t/\tau$ and $x_{t+1} \leftarrow x_t$, we recover $\varepsilon_{t+1} \leqslant 2\tau\rho|\phi_{t+1}|$. Furthermore, observe that this update is a geometric rescaling which ensures that BMP requires only $N_{\text{dual}} = \mathcal{O}(\ln(1/\varepsilon))$ dual steps (see proofs). Thus, the total number of negative calls, i.e., the number of iterations requiring a complete scan of the dictionary, is only $\mathcal{O}(\ln(1/\varepsilon))$. Therefore, for this and for the blending of steps, the duality gap estimates $|\phi_t|$ are the key to the speed-up realized by BMP.

Parameters. BMP involves three (hyper-)parameters $\eta > 0$, $\kappa \geqslant 1$, and $\tau > 1$ to be set before running the algorithm. The parameter η needs to be tuned carefully, as its value affects the criterion in Line 5 to promote either speed of convergence (e.g., $\eta \sim 0.1$, promoting full steps) or sparsity of the iterates (e.g., $\eta \sim 1000$, promoting constrained steps). In our experiments (Section 7.4 and Appendix B.4), we found that setting $\eta \sim 5$ leads to close to both maximal speed of convergence and sparsity of the iterates, with the default choices $\kappa = \tau = 2$. In this setting, BMP converges (much) faster than GMP and has it-

erates with sparsity very comparable to that of OMP, and therefore it is possible to enjoy both properties of speed and sparsity simultaneously. Note that the value of κ also impacts the range of values of η to which BMP is sensitive, since the criterion (Line 5) tests $\min_{v \in \mathcal{S}'_t} \langle v, \nabla f(x_t) \rangle \leqslant \phi_t / \eta$ while the weak-separation oracle asks for $v \in \mathcal{D}'$ such that $\langle v, \nabla f(x_t) \rangle \leqslant \phi_t / \kappa$. As always, in specific experiments, parameter tuning might further improve performance.

7.3.1 Convergence analysis

We start with the simpler case of smooth convex functions of order $\ell > 1$ (Theorem 7.7). Our main result is Theorem 7.8, which subsumes the case of strongly convex functions. To establish the convergence rates of GMP and OMP, [88] assume knowledge of an upper bound on $\sup\{\|x^*\|_{\mathcal{D}'}, \|x_0\|_{\mathcal{D}'}, \dots, \|x_T\|_{\mathcal{D}'}\}$ where $\|\cdot\|_{\mathcal{D}'} : x \in \mathcal{H} \mapsto \inf\{\rho > 0 \mid x \in \rho \operatorname{conv} \mathcal{D}'\}$ is the *atomic norm*. In [89], this is resolved by working with the atomic norm $\|\cdot\|_{\mathcal{D}'}$ instead of the Hilbert space induced norm $\|\cdot\|$ to, e.g., define smoothness and strong convexity of f and derive the proofs, but $\|\cdot\|_{\mathcal{D}'}$ itself can be difficult to derive in many applications. In contrast, we need neither the finiteness assumption nor to change the norm, however we assume f to be coercive to ensure feasibility of problem (7.1), a reasonably mild assumption.

Theorem 7.7 (Smooth convex case)**.** *Let $\mathcal{D} \subset \mathcal{H}$ be a dictionary such that $0 \in \operatorname{int} \operatorname{conv} \mathcal{D}'$ and let $f \colon \mathcal{H} \to \mathbb{R}$ be smooth of order $\ell > 1$, convex, and coercive. Then BMP (Algorithm 7.3) ensures that $f(x_t) - \min_{\mathcal{H}} f \leqslant \varepsilon$ for all $t \geqslant T$ where*

$$T = \mathcal{O}\left(\left(\frac{L}{\varepsilon}\right)^{1/(\ell-1)}\right).$$

Proof. Let $\varepsilon > 0$ and $T = N_{\text{dual}} + N_{\text{full}} + N_{\text{constrained}} \in \mathbb{N} \cup \{+\infty\}$ where N_{dual}, N_{full}, and $N_{\text{constrained}}$ are the number of dual steps (Line 13), full steps (Line 17), and constrained steps (Line 7) taken in total respectively. The objective f is continuous and coercive so

$\arg\min_{\mathcal{H}} f \neq \varnothing$. Let $\varepsilon_t = f(x_t) - \min_{\mathcal{H}} f$ for $t \in \mathbb{N}$. Similarly to [15], we introduce *epoch starts* at iteration $t = 0$ or any iteration immediately following a dual step. Our goal is to bound the number of epochs and the number of iterations within each epoch. Notice that $0 \leqslant \varepsilon_{t+1} \leqslant \varepsilon_t$ and $\phi_t \leqslant \phi_{t+1} \leqslant 0$ for $t \in \mathbb{N}$.

Let $x^* \in \arg\min_{\mathcal{H}} f$. The function f is coercive and $f(x_{t+1}) \leqslant f(x_t)$ for $t \in \mathbb{N}$, so by Fact 7.2 the sequence of iterates is bounded. Define $\rho = \sup_{t\in\mathbb{N}} \|x_t - x^*\| < +\infty$. Note that ρ is independent of T. Let $t \in \mathbb{N}$ be an iteration of the algorithm, and $v_t^{\text{FW}} \in \arg\min_{v\in\mathcal{D}'} \langle v, \nabla f(x_t)\rangle = \arg\min_{z\in\operatorname{conv}\mathcal{D}'} \langle z, \nabla f(x_t)\rangle$. We can assume that $f(x_t) > f(x^*)$ otherwise the iterates have already converged. By convexity, $\langle x^* - x_t, \nabla f(x_t)\rangle < 0$. Since $0 \in \operatorname{int}(\operatorname{conv}\mathcal{D}')$, there exists $r > 0$ such that $\mathcal{B}(0, r) \subset \operatorname{conv}\mathcal{D}'$. Thus, $\frac{r(x^*-x_t)}{2\|x^*-x_t\|} \in \operatorname{conv}\mathcal{D}'$ so

$$\min_{z\in\operatorname{conv}\mathcal{D}'} \langle z, \nabla f(x_t)\rangle = \langle v_t^{\text{FW}}, \nabla f(x_t)\rangle \leqslant \left\langle \frac{r(x^* - x_t)}{2\|x^* - x_t\|}, \nabla f(x_t) \right\rangle < 0,$$

i.e.,

$$\begin{aligned} \langle x_t - x^*, \nabla f(x_t)\rangle &\leqslant \frac{2\|x_t - x^*\|}{r} \langle v_t^{\text{FW}}, -\nabla f(x_t)\rangle \\ &\leqslant \frac{2\rho}{r} \langle v_t^{\text{FW}}, -\nabla f(x_t)\rangle. \end{aligned} \tag{7.3}$$

By convexity,

$$f(x_t) - f(x^*) \leqslant \langle x_t - x^*, \nabla f(x_t)\rangle,$$

so with (7.3),

$$\varepsilon_t \leqslant \frac{2\rho}{r} \langle v_t^{\text{FW}}, -\nabla f(x_t)\rangle. \tag{7.4}$$

Let t be a dual step (Line 13). Then the weak-separation oracle call (Line 11) yields

$\langle v_t^{\text{FW}}, \nabla f(x_t)\rangle \geqslant \phi_t$. By (7.4) and Line 15,

$$\varepsilon_t \leqslant \frac{2\rho}{r}|\phi_t| \tag{7.5}$$

$$= \frac{2\rho}{r}\frac{|\phi_0|}{\tau^{n_{\text{dual}}}}, \tag{7.6}$$

where n_{dual} is the number of dual steps taken before t. Therefore, by (7.6) and since $\tau > 1$,

$$N_{\text{dual}} \leqslant \left\lceil \log_\tau \left(\frac{2\rho|\phi_0|}{r\varepsilon} \right) \right\rceil. \tag{7.7}$$

If a full step is taken (Line 17), then the weak-separation oracle (Line 11) returns $v_t \in \mathcal{D}'$ such that $\langle v_t, \nabla f(x_t)\rangle \leqslant \phi_t/\kappa$. By smoothness and using Lemma 7.3 with $\overline{\ell} = \ell/(\ell - 1) > 1$,

$$\begin{aligned} f(x_{t+1}) &\leqslant \min_{\gamma \in \mathbb{R}_+} f(x_t + \gamma v_t) \\ &\leqslant \min_{\gamma \in \mathbb{R}_+} f(x_t) + \gamma\langle v_t, \nabla f(x_t)\rangle + \frac{L}{\ell}\gamma^\ell \|v_t\|^\ell \\ &= f(x_t) - \frac{\langle v_t, -\nabla f(x_t)\rangle^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1}\|v_t\|^{\overline{\ell}}} \\ &\leqslant f(x_t) - \frac{|\phi_t/\kappa|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1}(D_{\mathcal{D}'}/2)^{\overline{\ell}}}, \end{aligned}$$

where we used $\|v_t\| \leqslant D_{\mathcal{D}'}/2$ (by symmetry). Therefore, the primal progress is at least

$$f(x_t) - f(x_{t+1}) \geqslant \frac{2^{\overline{\ell}}|\phi_t|^{\overline{\ell}}}{\overline{\ell}\kappa^{\overline{\ell}} L^{\overline{\ell}-1} D_{\mathcal{D}'}^{\overline{\ell}}}. \tag{7.8}$$

Lastly, if a constrained step is taken (Line 7), then by smoothness and using Lemma 7.3

with $-\tilde{\nabla} f(x_t)$,

$$\begin{aligned}
f(x_{t+1}) &\leqslant \min_{\gamma \in \mathbb{R}_+} f(x_t - \gamma \tilde{\nabla} f(x_t)) \\
&\leqslant \min_{\gamma \in \mathbb{R}_+} f(x_t) - \gamma \langle \tilde{\nabla} f(x_t), \nabla f(x_t) \rangle + \frac{L}{\ell} \gamma^\ell \|\tilde{\nabla} f(x_t)\|^\ell \\
&= f(x_t) - \frac{\langle \tilde{\nabla} f(x_t), \nabla f(x_t) \rangle^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} \|\tilde{\nabla} f(x_t)\|^{\overline{\ell}}} \\
&= f(x_t) - \frac{\|\tilde{\nabla} f(x_t)\|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1}} \\
&\leqslant f(x_t) - \frac{|\langle v_t^{\text{FW-}\mathcal{S}}, \tilde{\nabla} f(x_t) \rangle|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} \|v_t^{\text{FW-}\mathcal{S}}\|^{\overline{\ell}}} \\
&\leqslant f(x_t) - \frac{|\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t) \rangle|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} \|v_t^{\text{FW-}\mathcal{S}}\|^{\overline{\ell}}} \\
&\leqslant f(x_t) - \frac{|\phi_t / \eta|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} (D_{\mathcal{D}'}/2)^{\overline{\ell}}},
\end{aligned}$$

where the last three lines respectively come from the Cauchy-Schwarz inequality, $v_t^{\text{FW-}\mathcal{S}} \in \operatorname{span} \mathcal{S}_t$, $\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t) \rangle \leqslant \phi_t / \eta$ (Line 5), and $\|v_t^{\text{FW-}\mathcal{S}}\| \leqslant D_{\mathcal{D}'}/2$ (by symmetry). Therefore, the primal progress is at least

$$f(x_t) - f(x_{t+1}) \geqslant \frac{2^{\overline{\ell}} |\phi_t|^{\overline{\ell}}}{\overline{\ell} \eta^{\overline{\ell}} L^{\overline{\ell}-1} D_{\mathcal{D}'}^{\overline{\ell}}}, \tag{7.9}$$

whose lower bound only differs by a constant factor $(\kappa/\eta)^{\overline{\ell}}$ from that of a full step (7.8).

Now, we have

$$\begin{aligned}
T &= N_{\text{dual}} + N_{\text{full}} + N_{\text{constrained}} \\
&= N_{\text{dual}} + \sum_{\substack{t=0 \\ t \text{ epoch start}}}^{T-1} \left(N_{\text{full}}^{(t)} + N_{\text{constrained}}^{(t)} \right),
\end{aligned} \tag{7.10}$$

where $N_{\text{full}}^{(t)}$ and $N_{\text{constrained}}^{(t)}$ are the number of full steps and constrained steps taken during epoch t respectively. Let $t > 0$ be an epoch start. Thus, $t - 1$ is a dual step. By (7.5), since

$x_t = x_{t-1}$ and $\phi_t = \phi_{t-1}/\tau$,

$$\varepsilon_t \leqslant \frac{2\rho\tau}{r}|\phi_t|. \tag{7.11}$$

This also holds for $t = 0$ by (7.4) and Line 2 (and actually for all $t \in [\![0, T]\!]$). By (7.8) and (7.9), since $\phi_s = \phi_t$ for all nondual steps s in the epoch starting at t,

$$\begin{aligned}\varepsilon_t &\geqslant \sum_{s\in\text{epoch}(t)} (f(x_s) - f(x_{s+1})) \\ &\geqslant \left(N_{\text{full}}^{(t)} + N_{\text{constrained}}^{(t)}\right) \frac{2^{\overline{\ell}}|\phi_t|^{\overline{\ell}}}{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}.\end{aligned} \tag{7.12}$$

Combining (7.11) and (7.12),

$$N_{\text{full}}^{(t)} + N_{\text{constrained}}^{(t)} \leqslant \frac{2\rho\tau}{r}\frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}|\phi_t|^{1-\overline{\ell}}. \tag{7.13}$$

Therefore, by (7.10), (7.13), and $\overline{\ell} > 1$,

$$\begin{aligned}T &\leqslant N_{\text{dual}} + \frac{2\rho\tau}{r}\frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}\sum_{t=0}^{N_{\text{dual}}}\left(\frac{|\phi_0|}{\tau^t}\right)^{1-\overline{\ell}} \\ &= N_{\text{dual}} + \frac{2\rho\tau}{r}\frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}|\phi_0|^{1-\overline{\ell}}\frac{\tau^{(\overline{\ell}-1)(N_{\text{dual}}+1)} - 1}{\tau^{\overline{\ell}-1} - 1}.\end{aligned}$$

By (7.7),

$$\begin{aligned}T &\leqslant \log_\tau\left(\frac{2\rho|\phi_0|}{r\varepsilon}\right) + 1 + \frac{2\rho\tau}{r}\frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}\frac{|\phi_0|^{1-\overline{\ell}}}{\tau^{\overline{\ell}-1} - 1}\left(\tau^{(\overline{\ell}-1)\left(\log_\tau\left(\frac{2\rho|\phi_0|}{r\varepsilon}\right)+2\right)} - 1\right) \\ &= \log_\tau\left(\frac{2\rho|\phi_0|}{r\varepsilon}\right) + 1 + \frac{2\rho\tau}{r}\frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}\frac{|\phi_0|^{1-\overline{\ell}}}{\tau^{\overline{\ell}-1} - 1}\left(\tau^{2(\overline{\ell}-1)}\left(\frac{2\rho|\phi_0|}{r\varepsilon}\right)^{\overline{\ell}-1} - 1\right).\end{aligned}$$

We conclude that the algorithm converges with

$$T = \mathcal{O}\left(\left(\frac{L}{\varepsilon}\right)^{1/(\ell-1)}\right).$$

□

We now present our main result in its full generality. We provide the general convergence rates of BMP (Algorithm 7.3) in Theorem 7.8. Recall that sharpness is implied by strong convexity and that it is a very mild assumption in finite dimensional spaces as it is satisfied by all *well-behaved* convex functions (Lemma 7.5).

Theorem 7.8 (Smooth convex sharp case)**.** *Let $\mathcal{D} \subset \mathcal{H}$ be a dictionary such that $0 \in \operatorname{int}\operatorname{conv}\mathcal{D}'$ and let $f\colon \mathcal{H} \to \mathbb{R}$ be L-smooth of order $\ell > 1$, convex, coercive, and σ-sharp of order $\theta \in]0, 1/\ell]$ on $\mathcal{K}$. Then BMP (Algorithm 7.3) ensures that $f(x_t) - \min_{\mathcal{H}} f \leqslant \varepsilon$ for all $t \geqslant T$ where*

$$T = \begin{cases} \mathcal{O}\left(\sigma^{1/(1-\theta)} L^{1/(\ell-1)} \ln\left(\dfrac{\sigma|\phi_0|}{\varepsilon^{1-\theta}}\right)\right) & \text{if } \ell\theta = 1 \\ \mathcal{O}\left(\left(\dfrac{\sigma^\ell L}{\varepsilon^{1-\ell\theta}}\right)^{1/(\ell-1)}\right) & \text{if } \ell\theta < 1. \end{cases}$$

Moreover, $\operatorname{dist}(x_t, \arg\min_{\mathcal{H}} f) \to 0$ *as* $t \to +\infty$ *at same rate.*

If f is not strongly convex then [88] only guarantee a sublinear convergence rate $\mathcal{O}(1/\varepsilon)$ for GMP and OMP, while Theorem 7.8 can still guarantee higher convergence rates, up to linear convergence $\mathcal{O}(\ln(1/\varepsilon))$ if $\ell\theta = 1$, using sharpness. Note that in the popular case of smooth strongly convex functions of orders $\ell = 2$ and $s = 2$, Theorem 7.8 guarantees a linear convergence rate as these functions are sharp of order $\theta = 1/2$ (with constant $C = \sqrt{2/S}$) and thus satisfy $\ell\theta = 1$. In conclusion, Theorem 7.8 extends linear convergence rates to a large class of non-strongly convex functions solving problem (7.1).

Proof. Let $\varepsilon > 0$. By Theorem 7.7, there exists $T \in \mathbb{N}$ such that $f(x_T) - \min_{\mathcal{H}} f \leqslant \varepsilon$. Let $\varepsilon_t = f(x_t) - \min_{\mathcal{H}} f$ for $t \in \mathbb{N}$ and $T = N_{\text{dual}} + N_{\text{full}} + N_{\text{constrained}}$ where N_{dual}, N_{full}, and

$N_{\text{constrained}}$ are the number of dual steps (Line 13), full steps (Line 17), and constrained steps (Line 7) taken in total respectively. Similarly to [15], we introduce *epoch starts* at iteration $t = 0$ or any iteration immediately following a dual step. Our goal is to bound the number of epochs and the number of iterations within each epoch. Notice that $0 \leqslant \varepsilon_{t+1} \leqslant \varepsilon_t$ and $\phi_t \leqslant \phi_{t+1} \leqslant 0$ for $t \in [\![0, T]\!]$.

Let $t \in [\![0, T]\!]$ be an iteration of the algorithm, $v_t^{\text{FW}} \in \arg\min_{v \in \mathcal{D}'} \langle v, \nabla f(x_t) \rangle$, and $x_t^* = \operatorname{proj}(x_t, \arg\min_{\mathcal{H}} f)$. We can assume that $f(x_t) > f(x_t^*)$ otherwise the iterates have already converged. By convexity, $\langle x_t^* - x_t, \nabla f(x_t) \rangle < 0$. Since $0 \in \operatorname{int} \operatorname{conv} \mathcal{D}'$, there exists $r > 0$ such that $\mathcal{B}(0, r) \subset \operatorname{conv} \mathcal{D}'$. Therefore, $\frac{r(x_t^* - x_t)}{2\|x_t^* - x_t\|} \in \operatorname{conv} \mathcal{D}'$ so

$$\min_{z \in \operatorname{conv} \mathcal{D}'} \langle z, \nabla f(x_t) \rangle = \langle v_t^{\text{FW}}, \nabla f(x_t) \rangle \leqslant \left\langle \frac{r(x_t^* - x_t)}{2\|x_t^* - x_t\|}, \nabla f(x_t) \right\rangle < 0,$$

i.e.,

$$\frac{r\langle x_t - x_t^*, \nabla f(x_t) \rangle}{-2\langle v_t^{\text{FW}}, \nabla f(x_t) \rangle} \leqslant \|x_t - x_t^*\|. \tag{7.14}$$

The sharpness of f implies that $\arg\min_{\mathcal{H}} f \subset \operatorname{int} \mathcal{K}$. Let $r_t^* \in]0, \|x_t - x_t^*\|[$ such that $\mathcal{B}(x_t^*, r_t^*) \subset \mathcal{K}$, and let $\rho \min\{r_0^*/\|x_0 - x_0^*\|, \ldots, r_T^*/\|x_T - x_T^*\|\} \in]0, 1[$. Then, $x_t^* + \rho(x_t - x_t^*) \in \mathcal{B}(x_t^*, r_t^*) \subset \mathcal{K}$. By convexity, $x_t^* = \operatorname{proj}(x_t^* + \rho(x_t - x_t^*), \arg\min_{\mathcal{H}} f)$: indeed, $\langle x_t^* - x^*, x_t - x_t^* \rangle \geqslant 0$ for all $x^* \in \arg\min_{\mathcal{H}} f$, thus

$$\begin{aligned} \|(x_t^* + \rho(x_t - x_t^*)) - x^*\|^2 &= \|x_t^* - x^*\|^2 + \rho^2 \|x_t - x_t^*\|^2 + 2\rho \langle x_t^* - x^*, x_t - x_t^* \rangle \\ &\geqslant \rho^2 \|x_t - x_t^*\|^2, \end{aligned} \tag{7.15}$$

where (7.15) is an equality if and only if $x^* = x_t^*$. Hence, using sharpness,

$$\begin{aligned}
\rho\|x_t - x_t^*\| &= \|(x_t^* + \rho(x_t - x_t^*)) - x_t^*\| \\
&\leqslant \sigma(f(x_t^* + \rho(x_t - x_t^*)) - f(x_t^*))^\theta \\
&\leqslant \sigma(f(x_t^*) + \rho(f(x_t) - f(x_t^*)) - f(x_t^*))^\theta \\
&= \sigma\rho^\theta(f(x_t) - f(x_t^*))^\theta \\
&\leqslant \sigma\rho^\theta\langle x_t - x_t^*, \nabla f(x_t)\rangle^\theta,
\end{aligned} \tag{7.16}$$

where the second and last inequalities come from convexity. Combining with (7.14), we get

$$\frac{r\langle x_t - x_t^*, \nabla f(x_t)\rangle}{-2\langle v_t^{\text{FW}}, \nabla f(x_t)\rangle} \leqslant \frac{\sigma}{\rho^{1-\theta}}\langle x_t - x_t^*, \nabla f(x_t)\rangle^\theta,$$

so, by convexity, we obtain the primal bound

$$f(x_t) - f(x_t^*) \leqslant \langle x_t - x_t^*, \nabla f(x_t)\rangle \leqslant \frac{1}{\rho}\left(-\frac{2C}{r}\langle v_t^{\text{FW}}, \nabla f(x_t)\rangle\right)^{1/(1-\theta)},$$

i.e.,

$$\varepsilon_t \leqslant \frac{1}{\rho}\left(-\frac{2C}{r}\langle v_t^{\text{FW}}, \nabla f(x_t)\rangle\right)^{1/(1-\theta)}. \tag{7.17}$$

Let t be a dual step (Line 13). Then the weak-separation oracle call (Line 11) yields $\langle v_t^{\text{FW}}, \nabla f(x_t)\rangle \geqslant \phi_t$. By (7.17) and Line 15,

$$\varepsilon_t \leqslant \frac{1}{\rho}\left(\frac{2C}{r}|\phi_t|\right)^{1/(1-\theta)} \tag{7.18}$$

$$= \frac{1}{\rho}\left(\frac{2C}{r}\frac{|\phi_0|}{\tau^{n_{\text{dual}}}}\right)^{1/(1-\theta)}, \tag{7.19}$$

where n_{dual} is the number of dual steps taken before t. Therefore, by (7.19) and since $\tau > 1$

and $\theta \in]0,1[$,

$$N_{\text{dual}} \leqslant \left\lceil \log_\tau \left(\frac{2\sigma|\phi_0|}{r\rho^{1-\theta}\varepsilon^{1-\theta}} \right) \right\rceil. \tag{7.20}$$

If a full step is taken (Line 17), then the weak-separation oracle (Line 11) returns $v_t \in \mathcal{D}'$ such that $\langle v_t, \nabla f(x_t)\rangle \leqslant \phi_t/\kappa$. By smoothness and using Lemma 7.3 and $\overline{\ell} = \ell/(\ell-1) > 1$,

$$\begin{aligned} f(x_{t+1}) &\leqslant \min_{\gamma\in\mathbb{R}_+} f(x_t + \gamma v_t) \\ &\leqslant \min_{\gamma\in\mathbb{R}_+} f(x_t) + \gamma\langle v_t, \nabla f(x_t)\rangle + \frac{L}{\ell}\gamma^\ell \|v_t\|^\ell \\ &= f(x_t) - \frac{\langle v_t, -\nabla f(x_t)\rangle^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1}\|v_t\|^{\overline{\ell}}} \\ &\leqslant f(x_t) - \frac{|\phi_t/\kappa|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1}(D_{\mathcal{D}'}/2)^{\overline{\ell}}}, \end{aligned}$$

where we used $\|v_t\| \leqslant D_{\mathcal{D}'}/2$ (by symmetry). Therefore, the primal progress is at least

$$f(x_t) - f(x_{t+1}) \geqslant \frac{2^{\overline{\ell}}|\phi_t|^{\overline{\ell}}}{\overline{\ell}\kappa^{\overline{\ell}} L^{\overline{\ell}-1} D_{\mathcal{D}'}^{\overline{\ell}}}. \tag{7.21}$$

Lastly, if a constrained step is taken (Line 7), then by smoothness and using Lemma 7.3,

$$\begin{aligned}
f(x_{t+1}) &\leqslant \min_{\gamma \in \mathbb{R}_+} f(x_t - \gamma \tilde{\nabla} f(x_t)) \\
&\leqslant \min_{\gamma \in \mathbb{R}_+} f(x_t) - \gamma \langle \tilde{\nabla} f(x_t), \nabla f(x_t) \rangle + \frac{L}{\ell} \gamma^\ell \|\tilde{\nabla} f(x_t)\|^\ell \\
&= f(x_t) - \frac{\langle \tilde{\nabla} f(x_t), \nabla f(x_t) \rangle^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} \|\tilde{\nabla} f(x_t)\|^{\overline{\ell}}} \\
&= f(x_t) - \frac{\|\tilde{\nabla} f(x_t)\|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1}} \\
&\leqslant f(x_t) - \frac{|\langle \tilde{\nabla} f(x_t), v_t^{\text{FW-}\mathcal{S}} \rangle|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} \|v_t^{\text{FW-}\mathcal{S}}\|^{\overline{\ell}}} \\
&= f(x_t) - \frac{|\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t) \rangle|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} \|v_t^{\text{FW-}\mathcal{S}}\|^{\overline{\ell}}} \\
&\leqslant f(x_t) - \frac{|\phi_t / \eta|^{\overline{\ell}}}{\overline{\ell} L^{\overline{\ell}-1} (D_{\mathcal{D}'}/2)^{\overline{\ell}}},
\end{aligned}$$

where the last three lines respectively come from the Cauchy-Schwarz inequality, $v_t^{\text{FW-}\mathcal{S}} \in \operatorname{span} \mathcal{S}_t$, $\langle v_t^{\text{FW-}\mathcal{S}}, \nabla f(x_t) \rangle \leqslant \phi_t / \eta$ (Line 5), and $\|v_t^{\text{FW-}\mathcal{S}}\| \leqslant D_{\mathcal{D}'}/2$ (by symmetry). Therefore, the primal progress is at least

$$f(x_t) - f(x_{t+1}) \geqslant \frac{2^{\overline{\ell}} |\phi_t|^{\overline{\ell}}}{\overline{\ell} \eta^{\overline{\ell}} L^{\overline{\ell}-1} D_{\mathcal{D}'}^{\overline{\ell}}}. \tag{7.22}$$

whose lower bound only differs by a constant factor $(\kappa/\eta)^{\overline{\ell}}$ from that of a full step (7.21).

Now, we have

$$\begin{aligned}
T &= N_{\text{dual}} + N_{\text{full}} + N_{\text{constrained}} \\
&= N_{\text{dual}} + \sum_{\substack{t=0 \\ t \text{ epoch start}}}^{T-1} \left(N_{\text{full}}^{(t)} + N_{\text{constrained}}^{(t)} \right),
\end{aligned} \tag{7.23}$$

where $N_{\text{full}}^{(t)}$ and $N_{\text{constrained}}^{(t)}$ are the number of full steps and constrained steps taken during epoch t respectively. Let $t > 0$ be an epoch start. Thus, $t - 1$ is a dual step. By (7.18),

since $x_t = x_{t-1}$ and $\phi_t = \phi_{t-1}/\tau$,

$$\varepsilon_t \leqslant \frac{1}{\rho}\left(\frac{2\tau\sigma}{r}|\phi_t|\right)^{1/(1-\theta)}. \tag{7.24}$$

This also holds for $t = 0$ by (7.17) and Line 2 (and actually for all $t \in [\![0, T]\!]$). By (7.21) and (7.22), since $\phi_s = \phi_t$ for all nondual steps s in the epoch starting at t,

$$\begin{aligned}\varepsilon_t &\geqslant \sum_{s\in\text{epoch}(t)} (f(x_s) - f(x_{s+1})) \\ &\geqslant \left(N_{\text{full}}^{(t)} + N_{\text{constrained}}^{(t)}\right) \frac{2^{\overline{\ell}}|\phi_t|^{\overline{\ell}}}{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}.\end{aligned} \tag{7.25}$$

Combining (7.24) and (7.25),

$$N_{\text{full}}^{(t)} + N_{\text{constrained}}^{(t)} \leqslant \frac{1}{\rho}\left(\frac{2\tau\sigma}{r}\right)^{1/(1-\theta)} \frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}|\phi_t|^{1/(1-\theta)-\overline{\ell}}. \tag{7.26}$$

Therefore, by (7.23) and (7.26),

$$T \leqslant N_{\text{dual}} + \frac{1}{\rho}\left(\frac{2\tau\sigma}{r}\right)^{1/(1-\theta)} \frac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}} \sum_{t=0}^{N_{\text{dual}}} \left(\frac{|\phi_0|}{\tau^t}\right)^{1/(1-\theta)-\overline{\ell}} \tag{7.27}$$

$$= \begin{cases} N_{\text{dual}} + \dfrac{1}{\rho}\left(\dfrac{2\tau\sigma}{r}\right)^{1/(1-\theta)} \dfrac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}(N_{\text{dual}}+1) & \\ & \text{if } \ell\theta = 1 \\ N_{\text{dual}} + \dfrac{1}{\rho}\left(\dfrac{2\tau\sigma}{r}\right)^{1/(1-\theta)} \dfrac{\overline{\ell}\max\{\kappa^{\overline{\ell}}, \eta^{\overline{\ell}}\}L^{\overline{\ell}-1}D_{\mathcal{D}'}^{\overline{\ell}}}{2^{\overline{\ell}}}|\phi_0|^{1/(1-\theta)-\overline{\ell}} \dfrac{\left(\tau^{\overline{\ell}-1/(1-\theta)}\right)^{N_{\text{dual}}+1} - 1}{\tau^{\overline{\ell}-1/(1-\theta)} - 1} & \\ & \text{if } \ell\theta < 1 \end{cases}$$

where, if $\alpha = \frac{1-\ell\theta}{(\ell-1)(1-\theta)} = \overline{\ell} - \frac{1}{1-\theta}$, by (7.20) we have

$$\begin{aligned}
\left(\tau^{\overline{\ell}-1/(1-\theta)}\right)^{N_{\text{dual}}+1} &= \left(\tau^{\alpha}\right)^{N_{\text{dual}}+1} \\
&= \exp\left(\alpha \ln(\tau)(N_{\text{dual}}+1)\right) \\
&\leqslant \exp\left(\alpha \ln(\tau)\left(\log_{\tau}\left(\frac{2\sigma|\phi_0|}{r\rho^{1-\theta}\varepsilon^{1-\theta}}\right)+2\right)\right) \\
&= \exp\left(\alpha \ln\left(\frac{2\sigma|\phi_0|}{r\rho^{1-\theta}\varepsilon^{1-\theta}}\right) + 2\alpha\ln(\tau)\right) \\
&= \tau^{2\alpha}\left(\frac{2\sigma|\phi_0|}{r\rho^{1-\theta}\varepsilon^{1-\theta}}\right)^{\alpha}.
\end{aligned}$$

By (7.20), we conclude that

$$T = \begin{cases} \mathcal{O}\left(\sigma^{1/(1-\theta)} L^{1/(\ell-1)} \ln\left(\frac{\sigma|\phi_0|}{\varepsilon^{1-\theta}}\right)\right) & \text{if } \ell\theta = 1 \\ \mathcal{O}\left(\left(\frac{\sigma^{\ell} L}{\varepsilon^{1-\ell\theta}}\right)^{1/(\ell-1)}\right) & \text{if } \ell\theta < 1. \end{cases}$$

Finally, by (7.16),

$$\|x_t - x_t^*\| \leqslant \frac{\sigma}{\rho^{1-\theta}}\varepsilon_t^{\theta}$$

for all $t \in \mathbb{N}$. Thus, $\|x_t - x_t^*\| \to 0$ as $t \to +\infty$. □

Remark 7.9 (Optimality of the convergence rates)**.** *Let $n \leqslant +\infty$ be the dimension of $\mathcal{H}$. Lower bounds are provided in [107] when solving problem* (7.1) *in different cases. These optimal rates are reported in Table 7.1, where we compare them to those of BMP proved in this chapter (Theorems 7.7–7.8). The third column gives the lower bounds on complexity stated in [107, Eq. (1.20), (1.21'), and (1.21)]. Note that our rates are dimension independent and hold globally across iterations. It remains an open question to determine whether the gap in the exponent can be closed by accelerating BMP.*

Table 7.1: Comparison of the rates of BMP vs. the lower bounds on complexity.

Properties of f	**BMP rate**	**Lower bound on complexity**
Smooth convex	$T(\varepsilon) = \mathcal{O}\left(\frac{1}{\varepsilon^{1/(\ell-1)}}\right)$	$T(\varepsilon) = \Omega\left(\min\left\{n, \frac{1}{\varepsilon^{1/(1.5\ell-1)}}\right\}\right)$
Smooth convex sharp with $\ell = 2$, $\theta = 1/2$	$T(\varepsilon) = \mathcal{O}\left(\ln\left(\frac{1}{\varepsilon}\right)\right)$	$T(\varepsilon) = \Omega\left(\min\left\{n, \ln\left(\frac{1}{\varepsilon}\right)\right\}\right)$
Smooth convex sharp with $\ell\theta < 1$	$T(\varepsilon) = \mathcal{O}\left(\frac{1}{\varepsilon^{(1-\ell\theta)/(\ell-1)}}\right)$	$T(\varepsilon) = \Omega\left(\min\left\{n, \frac{1}{\varepsilon^{(1-\ell\theta)/(1.5\ell-1)}}\right\}\right)$

7.4 Computational experiments

We implemented BMP in Python 3 along with GMP and OMP, the Accelerated Matching Pursuit algorithm (accMP) [89], and the Blended Conditional Gradient (BCG) and Conditional Gradient with Enhancement and Truncation (CoGEnT) [116] algorithms for completeness. All algorithms share the same code framework to ensure fair comparison both in iteration and wall-clock time performance, as well as for sparsity analysis. No enhancement beyond basic coding was performed. We ran the experiments on a laptop under Linux Ubuntu 18.04 with Intel Core i7 3.5GHz CPU and 8GB RAM. The code is available at https://github.com/cyrillewcombettes/bmp. The random data are drawn from Gaussian distributions. For GMP, OMP, BCG, and CoGEnT, we represented the duality gaps by $-\min_{v\in\mathcal{D}'}\langle v, \nabla f(x_t)\rangle$, yielding a zig-zaging plot dissimilar to the stair-like plot of the duality gap estimates $|\phi_t|$ of BMP. Appendix B.4 contains additional experiments.

7.4.1 Comparison of BMP vs. GMP, OMP, BCG, and CoGEnT

Let $\mathcal{H}$ be the Euclidean space $(\mathbb{R}^n, \langle\cdot,\cdot\rangle)$ and $\mathcal{D}$ be the set of signed canonical vectors $\{\pm e_1, \ldots, \pm e_n\}$. Suppose we want to learn the (sparse) source x^* from observed data $y = Ax^* + w$, where $A \in \mathbb{R}^{m\times n}$ and where $w \sim \mathcal{N}(0, \sigma^2 I_m)$ is the noise in the observed

y. The general and most intuitive formulation of the problem is:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \|y - Ax\|_2^2 \\ &\text{s.t. } \|x\|_0 \leqslant \|x^*\|_0 s \end{aligned}$$

but the ℓ_0-pseudo norm constraint $\|\cdot\|_0 : x \in \mathbb{R}^n \mapsto \operatorname{card}\{i \in [\![1, n]\!] \mid \langle x, e_i\rangle \neq 0\}$ is non-convex and makes the problem NP-hard and therefore intractable in many situations [103]. To remedy this, this sparsity constraint can be handled in various ways, either by completely removing it and relying on an algorithm inherently promoting sparsity, or through a convex relaxation of the constraint, often via the ℓ_1-norm, and then solving the new constrained convex problem. BMP, GMP, and OMP follow the first option and solve the unconstrained (and unregularized) problem:

$$\min_{x\in\mathbb{R}^n} \|y - Ax\|_2^2.$$

On the other hand, BCG and CoGEnT follow the second option and solve the relaxed constrained problem:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \|y - Ax\|_2^2 \\ &\text{s.t. } \|x\|_1 \leqslant \|x^*\|_1. \end{aligned}$$

We ran a comparison of these methods, where we favorably provided the constraint $\|x\|_1 \leqslant \|x^*\|_1$ for BCG and CoGEnT although x^* is unknown. We set $m = 500$, $n = 2000$, $s = 100$, and $\sigma = 0.05$. In BMP, we set $\kappa = \tau = 2$ and we chose $\eta = 5$; see Appendix B.4.1 for an in-depth sensitivity analysis of BMP with respect to η. We did not perform any additional correction of the active sets (Line 22). Note that [116, Tab. III] demonstrated the superiority of CoGEnT over CoSaMP [104], Subspace Pursuit [31], and Gradient Descent with Sparsification [48] on an equivalent experiment and we therefore do not compare to

those methods.

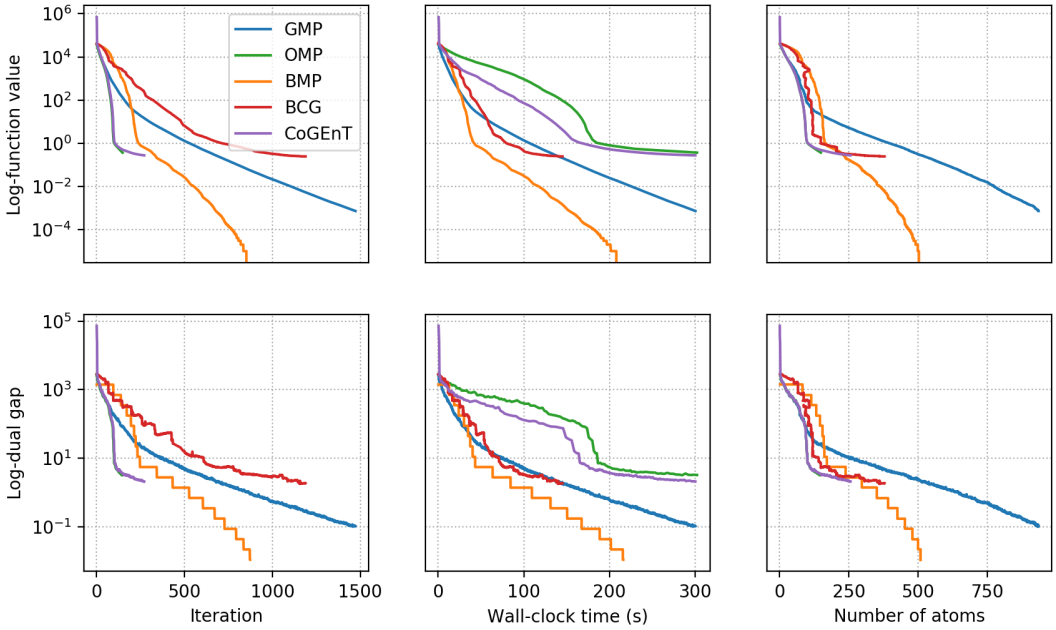

Figure 7.1: Comparison of BMP vs. GMP, OMP, BCG, and CoGEnT, with $\eta = 5$.

Figure 7.1 shows that BMP is the fastest algorithm in wall-clock time and has close-to-optimal sparsity. It is important to stress that, unlike BCG and CoGEnT, BMP achieves this while having no explicit sparsity-promoting constraint, regularization, nor information on x^*. Thus, when $\|x^*\|_1$ is not provided, which is the case in most applications, BCG and CoGEnT would require a hyper-parameter tuning of the sparsity-inducing constraint (or, equivalently, the Lagrangian penalty parameters), such as the radius of the ℓ_1-ball [124], as used here, or the trace-norm-ball [42]. OMP and CoGEnT converge faster per-iteration, as expected, given that they solve a reoptimization problem at each iteration, however this is very costly and the disadvantage becomes evident in wall-clock time performance. Note that another "obvious" choice for an algorithm would be projected gradient descent, however the provided sparsity is far from sufficient; see Appendix B.4.2.

In Figure 7.2, we compare the Normalized Mean Squared Error (NMSE) of the different methods. The NMSE at iterate x_t is defined as $\|x_t - x^*\|_2^2 / \|x^*\|_2^2$. The plots show a rebound occurring once the NMSE reaches $\sim 10^{-4}$, which is due to the algorithms overfitting to the

noisy measurements y. A post-processing step can mitigate the rebound via early stopping or by removing atoms whose coefficient in the decomposition of the iterate are smaller than some threshold $\delta > 0$.

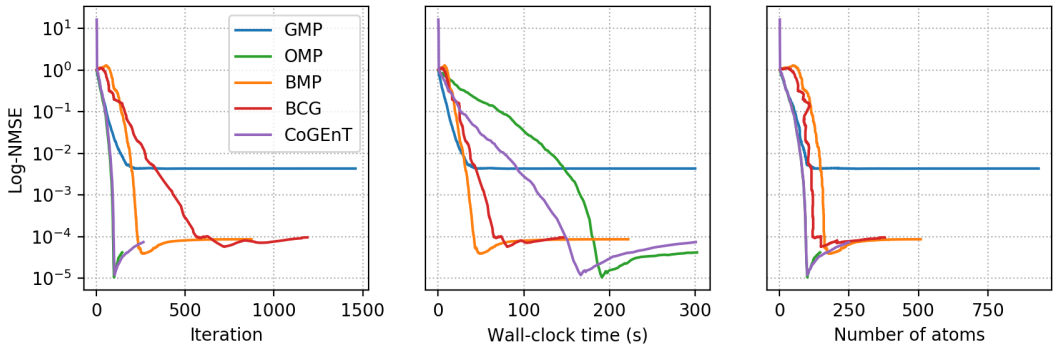

Figure 7.2: Comparison in NMSE of BMP vs. GMP, OMP, BCG, and CoGEnT, with $\eta = 5$.

We used early stopping on a validation set and present the test error $\|y_{\text{test}} - A_{\text{test}} x_T\|_2^2 / m_{\text{test}}$ on a test set in Tab. 7.2, where x_T is the solution iterate for each algorithm. For completeness, we also reported the results for the Gradient Hard Thresholding Pursuit (GraHTP) and Fast Gradient Hard Thresolding Pursuit (Fast GraHTP) algorithms [133], for which we favorably set $k = \|x^*\|_0$. As expected, GMP performs the worst on the test set because its NMSE does not achieve sufficient convergence (see Figure 7.2), highlighting the importance of a clean, i.e., sparse, decomposition into the dictionary $\mathcal{D}$.

Table 7.2: Test error achieved using early stopping on a validation set.

Algorithm	GMP	OMP	BMP	BCG	CoGEnT	GraHTP	Fast GraHTP
Test error	0.1917	0.0036	0.0037	0.0068	0.0043	0.0036	0.0037

Appendix B.4 contains additional experiments on different objective functions: an arbitrarily chosen norm (Appendix B.4.3), the Huber loss (Appendix B.4.4), the distance to a convex set (Appendix B.4.5), and a logistic regression loss (Appendix B.4.6). The conclusions are identical.

7.4.2 Comparison of BMP vs. accMP

An Accelerated Matching Pursuit algorithm (accMP) for solving problem (7.1) was recently proposed in [89]. We implemented the same code as theirs, using the exact same parametrization. The code framework matches the one we used for BMP. We ran BMP on their toy data example and compared the results against accMP (which they labeled *accelerated steepest* in their plot); notice that we recovered their (per-iteration) plot exactly. The experiment is to minimize $f\colon x \in \mathbb{R}^{100} \mapsto \|x - b\|_2^2/2$ over the linear span of $\mathcal{D}$, where $\mathcal{D}$ is dictionary of 200 randomly chosen atoms in $\mathbb{R}^{100}$ and $b \in \mathbb{R}^{100}$ is also randomly chosen. The parameters of accMP, kindly provided by the authors of [89], were $L = 1000$ and $\nu = 1$. As before we did not perform any additional correction of the active sets (Line 22) for BMP. We report the results in Figure 7.3.

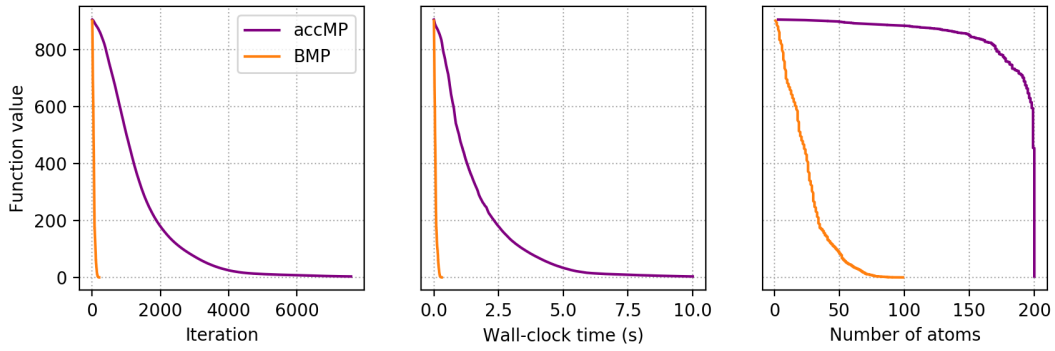

Figure 7.3: Comparison of BMP vs. accMP, with $\eta = 3$.

We see that BMP outperforms accMP in both speed of convergence and sparsity of the iterates. In fact, in terms of sparsity, accMP needs to use all available atoms to converge while BMP needs only half as much. Furthermore, accMP needs $\sim 75\%$ of all available atoms to start converging significantly while BMP starts to converge instantaneously. We suspect that this is due to the following: accMP accelerates coordinate descent-like directions, which might be relatively bad approximations of the actual descent direction $-\nabla f(x_t)$, whereas BMP is working directly with (the projection of) $-\nabla f(x_t)$, achieving much more progress and offsetting the effect of acceleration.

7.5 Final remarks

We presented a Blended Matching Pursuit algorithm (BMP) which enjoys both properties of fast rate of convergence and sparsity of the iterates. More specifically, we derived linear convergence rates for a large class of non-strongly convex functions solving problem (7.1), and we showed that our blending approach outperforms the state-of-the-art methods in speed of convergence while achieving close-to-optimal sparsity, and this without requiring sparsity-inducing constraints nor regularization. Although BMP already outperforms the Accelerated Matching Pursuit algorithm [89] in our experiments, we believe it is also amenable to acceleration.

Appendices

APPENDIX A

COMPLEMENTARY DEVELOPMENTS

A.1 AdamSFW: AdaSFW with momentum

Algorithm A.1 AdamSFW

Input: Start point $x_0 \in \mathcal{C}$, batch-size strategy $(b_t)_{t\in\mathbb{N}} \subset \mathbb{N}\backslash\{0\}$, momentum parameters $\beta_m, \beta_s \in]0,1[$, offset $\delta > 0$, number of inner iterations $K \in \mathbb{N}\backslash\{0\}$, learning rate $\eta > 0$.

1: $m_{-1}, s_{-1}, \bar{s}_{-1} \leftarrow 0, 0, 0$
2: **for** $t = 0$ **to** $T-1$ **do**
3: $\quad i_1, \ldots, i_b \overset{\text{i.i.d.}}{\sim} \mathcal{U}([\![1, m]\!])$
4: $\quad \tilde{\nabla} f(x_t) \leftarrow \dfrac{1}{b}\displaystyle\sum_{j=1}^{b} \nabla f_{i_j}(x_t)$
5: $\quad m_t \leftarrow \beta_m m_{t-1} + (1-\beta_m)\tilde{\nabla} f(x_t)$
6: $\quad s_t \leftarrow \beta_s s_{t-1} + (1-\beta_s)\tilde{\nabla} f(x_t)^2$
7: $\quad \bar{s}_t \leftarrow \max\{\bar{s}_{t-1}, s_t\}$
8: $\quad H_t \leftarrow \operatorname{diag}(\delta 1 + \sqrt{\bar{s}_t})$
9: $\quad y_0^{(t)} \leftarrow x_t$
10: $\quad$ **for** $k = 0$ **to** $K-1$ **do**
11: $\quad\quad \nabla Q_t(y_k^{(t)}) \leftarrow m_t + \dfrac{1}{\eta} H_t(y_k^{(t)} - x_t)$
12: $\quad\quad v_k^{(t)} \leftarrow \arg\min_{v\in\mathcal{C}} \langle v, \nabla Q_t(y_k^{(t)})\rangle$
13: $\quad\quad \gamma_k^{(t)} \leftarrow \min\left\{\eta \dfrac{\langle y_k^{(t)} - v_k^{(t)}, \nabla Q_t(y_k^{(t)})\rangle}{\|y_k^{(t)} - v_k^{(t)}\|_{H_t}^2}, 1\right\}$
14: $\quad\quad y_{k+1} \leftarrow y_k^{(t)} + \gamma_k^{(t)}(v_k^{(t)} - y_k^{(t)})$
15: $\quad$ **end for**
16: $\quad x_{t+1} \leftarrow y_K^{(t)}$
17: **end for**

In Algorithm A.1, inspired by Adam [71] and AMSGrad [117], we propose a variant of AdaSFW (Algorithm 5.4) with momentum which we used in our neural network training experiments (Section 5.5.2). The batch-size $b \in \mathbb{N}\backslash\{0\}$ and the learning rate $\eta > 0$ could be chosen as time-varying. Following [117], we require $\beta_m < \sqrt{\beta_s}$, with default values $\beta_m \leftarrow 0.9$ and $\beta_s \leftarrow 0.99$ or $\beta_s \leftarrow 0.999$. All operations in Line 8 are entrywise in

$\mathbb{R}^n$. Notice the presence of the momentum term m_t instead of $\tilde{\nabla} f(x_t)$ in Line 11: the subproblem addressed by AdamSFW is

$$\min_{x \in \mathcal{C}} \left\{ Q_t(x) = f(x_t) + \langle x - x_t, m_t \rangle + \frac{1}{2\eta} \|x - x_t\|_{H_t}^2 \right\}.$$

A.2 An application of the Douglas-Rachford algorithm

Let $\mathcal{H}$ be a Euclidean space with norm $\|\cdot\|$ and denote by $\Gamma_0(\mathcal{H})$ the set of proper lower semicontinuous convex functions $\mathcal{H} \to \mathbb{R} \cup \{+\infty\}$. The Douglas-Rachford algorithm [87] can be used to solve

$$\min_{x \in \mathcal{H}} f(x) + g(x),$$

when $f, g \in \Gamma_0(\mathcal{H})$ satisfy $\arg\min_{\mathcal{H}}(f+g) \neq \varnothing$ and $(\operatorname{ri} \operatorname{dom} f) \cap (\operatorname{ri} \operatorname{dom} g) \neq \varnothing$ [6], where ri and dom denote the relative interior of a set and the domain of a function respectively. It is presented in Algorithm A.2. For every function $\varphi \in \Gamma_0(\mathcal{H})$, the proximity operator is $\operatorname{prox}_\varphi = \arg\min_{x \in \mathcal{H}} \varphi(x) + (1/2)\|\cdot - x\|^2$ [101].

Algorithm A.2 Douglas-Rachford

Input: Start point $z_0 \in \mathcal{H}$.

1: **for** $t = 0$ **to** $T - 1$ **do**
2: $\quad x_t \leftarrow \operatorname{prox}_g(z_t)$
3: $\quad z_{t+1} \leftarrow \operatorname{prox}_f(2x_t - z_t) + z_t - x_t$
4: **end for**

We are interested in an application to problem (3.11), where $\mathcal{H} = \mathbb{R}^{n^2}$, $f = \iota_{\mathcal{K}} + (1/2)\|\cdot - y\|_2^2$, $g = \iota_{\mathcal{A}} + (1/2)\|\cdot - y\|_2^2$, $y \in \mathcal{H}$, $\mathcal{K} = \{x \in \mathbb{R}^{n^2} \mid x \geqslant 0\}$, $\mathcal{A} = \{x \in \mathbb{R}^{n^2} \mid Ax = 1\}$, and $A \in \mathbb{R}^{2n \times n^2}$ is defined in (3.10). Problem (3.11) admits a (unique) solution since it is a projection problem onto the intersection of the closed convex sets $\mathcal{K}$ and $\mathcal{A}$, and $1/n \in (\operatorname{ri} \operatorname{dom} f) \cap (\operatorname{ri} \operatorname{dom} g) = (\operatorname{ri} \mathcal{K}) \cap \mathcal{A}$ so the Douglas-Rachford algorithm is well

defined here. We now show that it reduces to Algorithm 3.2. For all $t \in \mathbb{N}$,

$$\begin{aligned}
x_t &= \operatorname{prox}_g(z_t) \\
&= \operatorname*{arg\,min}_{x \in \mathcal{H}} g(x) + \frac{1}{2}\|z_t - x\|_2^2 \\
&= \operatorname*{arg\,min}_{x \in \mathcal{A}} \frac{1}{2}\|x - y\|_2^2 + \frac{1}{2}\|x - z_t\|_2^2 \\
&= \operatorname*{arg\,min}_{x \in \mathcal{A}} \left\| x - \frac{z_t + y}{2} \right\|_2^2 \\
&= \operatorname{proj}\left(\frac{z_t + y}{2}, \mathcal{A} \right) \\
&= \frac{z_t + y}{2} - A^\dagger A \left(\frac{z_t + y}{2} - u \right),
\end{aligned}$$

since $\operatorname{proj}(\cdot, \mathcal{A}) = \cdot - A^\dagger A(\cdot - u)$, given any $u \in \mathcal{A}$ [6, Ex. 29.17]. Thus, Lines 2–3 in Algorithm 3.2 are equivalent to Line 2 in Algorithm A.2. Similarly, Line 4 in Algorithm 3.2 is equivalent to Line 3 in Algorithm A.2:

$$\begin{aligned}
z_{t+1} &= \operatorname{prox}_f(2x_t - z_t) + z_t - x_t \\
&= \operatorname*{arg\,min}_{z \in \mathcal{H}} \left(f(z) + \frac{1}{2}\|2x_t - z_t - z\|_2^2 \right) + z_t - x_t \\
&= \operatorname*{arg\,min}_{z \in \mathcal{K}} \left(\frac{1}{2}\|z - y\|_2^2 + \frac{1}{2}\|z - (2x_t - z_t)\|_2^2 \right) + z_t - x_t \\
&= \operatorname*{arg\,min}_{z \in \mathcal{K}} \left\| z - \frac{2x_t - z_t + y}{2} \right\|_2^2 + z_t - x_t \\
&= \operatorname{proj}\left(\frac{2x_t - z_t + y}{2}, \mathcal{K} \right) + z_t - x_t \\
&= \max\left\{ \frac{2x_t - z_t + y}{2}, 0 \right\} + z_t - x_t,
\end{aligned}$$

since $\operatorname{proj}(\cdot, \mathcal{K}) = \max\{\cdot, 0\}$.

APPENDIX B

COMPLEMENTARY PLOTS

B.1 Tables

We report the CPU times of Figures 3.1–3.2 in Tables B.1–B.3. In each table, the last column presents the statistics of the ratio of the CPU time for a projection to the CPU time for a linear minimization.

Table B.1: CPU times in Figure 3.1.

Dimension n	**CPU time (s)**		
	Linear minimization	Projection	Ratio
100	$4.27 \cdot 10^{-5}$ $\pm 2.0 \cdot 10^{-5}$	$1.30 \cdot 10^{-4}$ $\pm 6.3 \cdot 10^{-5}$	4.02 ± 3.1
1 000	$1.99 \cdot 10^{-5}$ $\pm 7.2 \cdot 10^{-6}$	$3.02 \cdot 10^{-4}$ $\pm 6.7 \cdot 10^{-5}$	16.8 ± 5.9
10 000	$3.02 \cdot 10^{-5}$ $\pm 2.6 \cdot 10^{-6}$	$2.17 \cdot 10^{-3}$ $\pm 2.1 \cdot 10^{-4}$	72.5 ± 9.6
100 000	$2.53 \cdot 10^{-4}$ $\pm 1.2 \cdot 10^{-4}$	$1.46 \cdot 10^{-2}$ $\pm 2.2 \cdot 10^{-3}$	68.5 ± 22
1 000 000	$1.80 \cdot 10^{-3}$ $\pm 6.4 \cdot 10^{-5}$	$1.37 \cdot 10^{-1}$ $\pm 1.2 \cdot 10^{-2}$	76.3 ± 6.4
10 000 000	$1.85 \cdot 10^{-2}$ $\pm 9.6 \cdot 10^{-4}$	$1.33 \cdot 10^{0}$ $\pm 1.2 \cdot 10^{-2}$	72.1 ± 3.5
100 000 000	$1.71 \cdot 10^{-1}$ $\pm 1.2 \cdot 10^{-3}$	$1.41 \cdot 10^{1}$ $\pm 3.6 \cdot 10^{-1}$	82.4 ± 2.1

Table B.2: CPU times in Figure 3.2 with a random input.

Dimension n	**CPU time (s)**		
	Linear minimization	Projection	Ratio
100	$1.13 \cdot 10^{-2} \pm 5.4 \cdot 10^{-3}$	$1.51 \cdot 10^{-2} \pm 9.1 \cdot 10^{-3}$	2.50 ± 3.3
200	$1.97 \cdot 10^{-2} \pm 2.4 \cdot 10^{-3}$	$3.48 \cdot 10^{-2} \pm 2.6 \cdot 10^{-3}$	1.77 ± 0.14
500	$4.56 \cdot 10^{-2} \pm 2.4 \cdot 10^{-3}$	$2.09 \cdot 10^{-1} \pm 2.1 \cdot 10^{-2}$	4.62 ± 0.71
1 000	$1.24 \cdot 10^{-1} \pm 1.2 \cdot 10^{-2}$	$1.09 \cdot 10^{0} \pm 2.6 \cdot 10^{-2}$	8.83 ± 0.81
2 000	$1.45 \cdot 10^{0} \pm 7.9 \cdot 10^{-2}$	$1.36 \cdot 10^{1} \pm 1.8 \cdot 10^{-1}$	9.43 ± 0.48
5 000	$1.97 \cdot 10^{1} \pm 2.0 \cdot 10^{0}$	$2.63 \cdot 10^{2} \pm 1.5 \cdot 10^{0}$	13.4 ± 1.3
10 000	$8.63 \cdot 10^{1} \pm 1.7 \cdot 10^{0}$	$1.96 \cdot 10^{3} \pm 2.0 \cdot 10^{1}$	22.8 ± 0.64

Table B.3: CPU times in Figure 3.2 with a random symmetric input.

Dimension n	**CPU time (s)**		
	Linear minimization	Projection	Ratio
100	$5.69 \cdot 10^{-2} \pm 2.4 \cdot 10^{-2}$	$1.33 \cdot 10^{-2} \pm 1.1 \cdot 10^{-3}$	0.414 ± 0.44
200	$7.27 \cdot 10^{-2} \pm 2.4 \cdot 10^{-3}$	$3.94 \cdot 10^{-2} \pm 3.0 \cdot 10^{-3}$	0.542 ± 0.033
500	$7.41 \cdot 10^{-2} \pm 1.6 \cdot 10^{-3}$	$2.09 \cdot 10^{-1} \pm 1.1 \cdot 10^{-2}$	2.82 ± 0.19
1 000	$9.92 \cdot 10^{-2} \pm 1.3 \cdot 10^{-3}$	$1.13 \cdot 10^{0} \pm 1.9 \cdot 10^{-2}$	11.4 ± 0.17
2 000	$4.21 \cdot 10^{-1} \pm 7.6 \cdot 10^{-3}$	$1.38 \cdot 10^{1} \pm 1.2 \cdot 10^{-1}$	32.7 ± 0.74
5 000	$2.54 \cdot 10^{0} \pm 1.8 \cdot 10^{-2}$	$2.63 \cdot 10^{2} \pm 1.1 \cdot 10^{0}$	103 ± 0.73
10 000	$9.63 \cdot 10^{0} \pm 7.9 \cdot 10^{-2}$	$1.98 \cdot 10^{3} \pm 1.7 \cdot 10^{1}$	206 ± 3.2

B.2 Comparisons in duality gap for BoostFW

We provide additional plots for each experiment of Section 4.4: comparisons in number of oracle calls and in duality gap. We did not account for the CPU time taken to plot the

duality gap. In number of oracle calls, the plots have a stair-like behavior as multiple calls can be made within an iteration. We see that BoostFW performs more oracle calls than the other methods in general however it converges faster both per iteration and in CPU time. Note that in the traffic assignment experiment (Figure B.3), BoostFW also converges faster per oracle call. In the sparse logistic regression experiment (Figure B.2), the line search-free strategies converge faster in CPU time and the line search strategies converge faster per iteration, but in the collaborative filtering experiment (Figure B.4), BoostFW-ls and BoostFW-L respectively converge faster than expected.

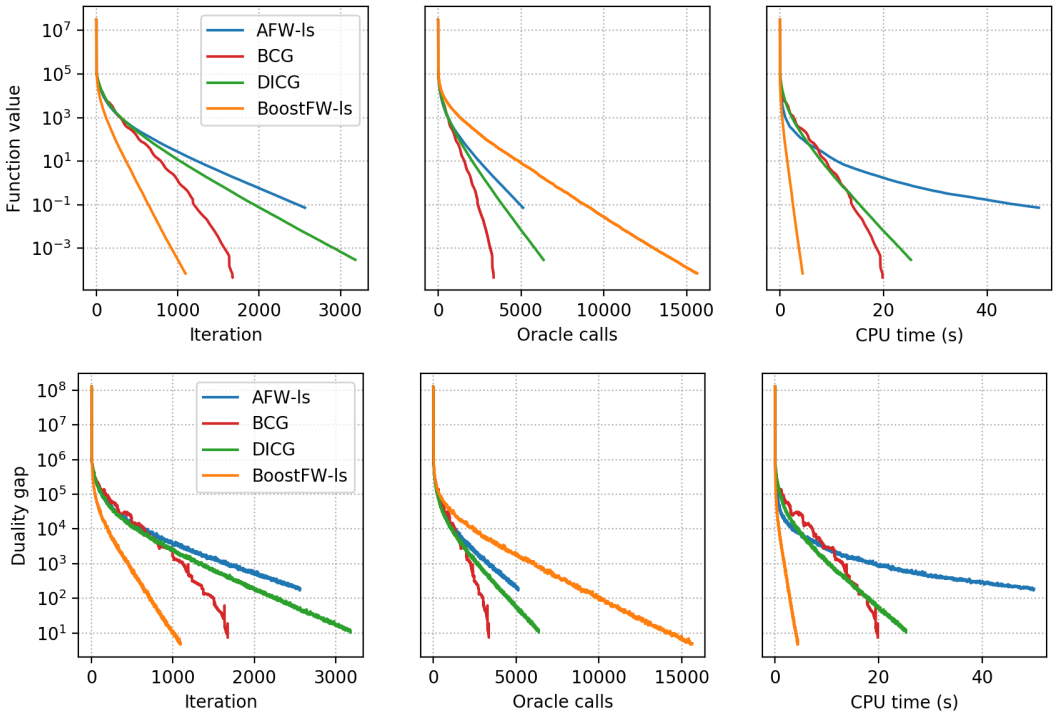

Figure B.1: Sparse signal recovery (Section 4.4.2).

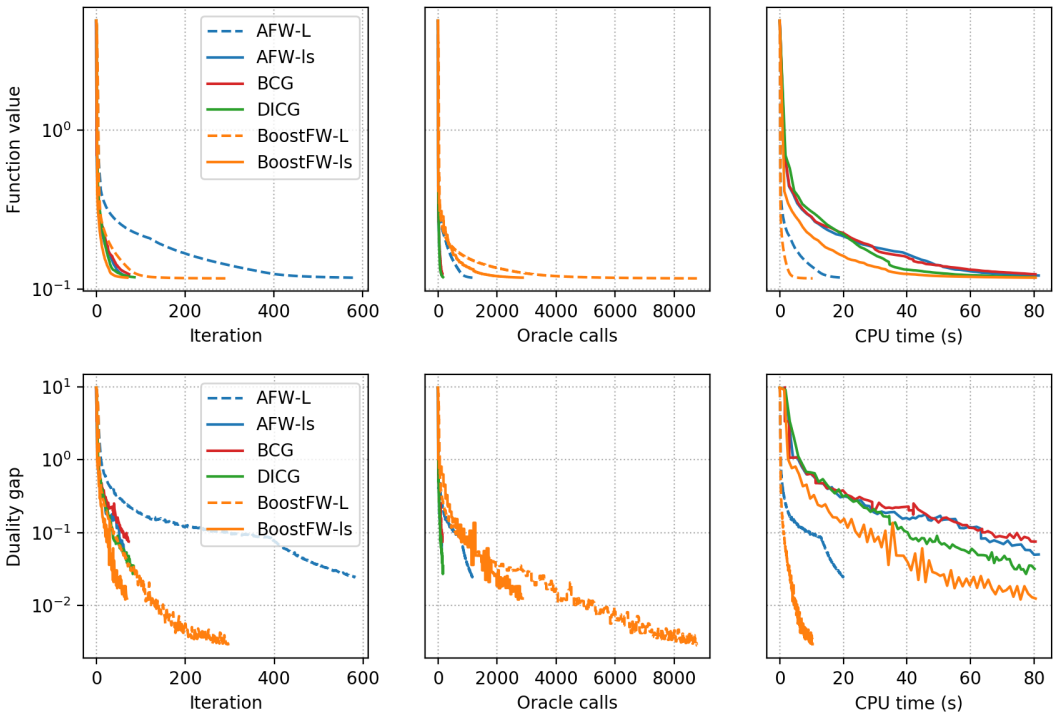

Figure B.2: Sparse logistic regression on the Gisette dataset (Section B.4.6).

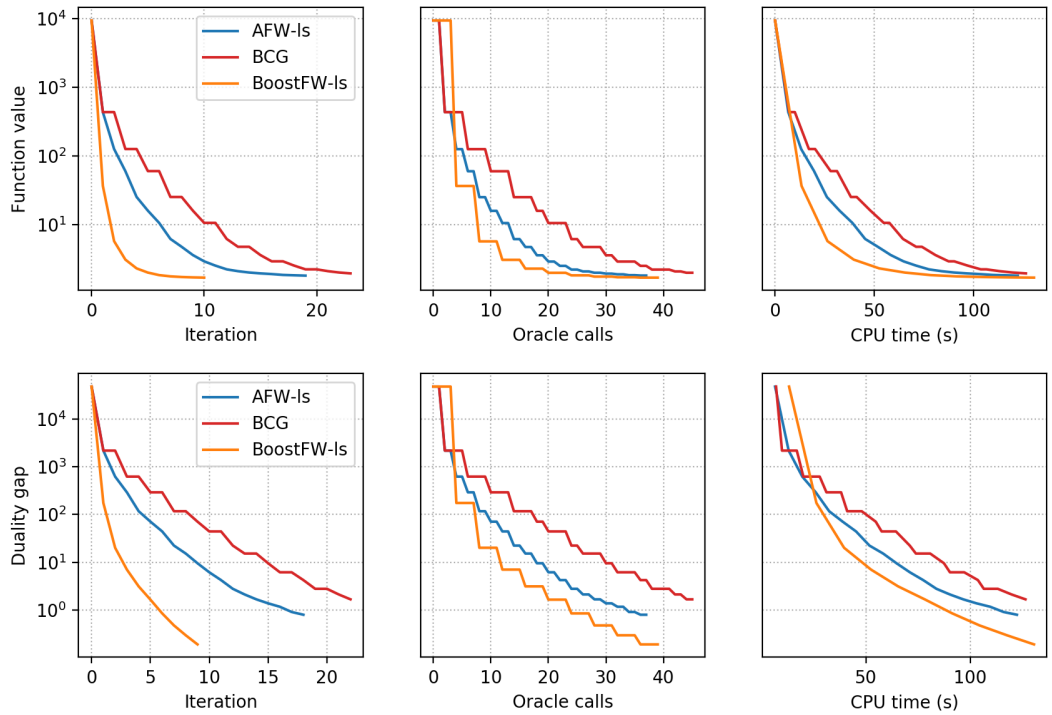

Figure B.3: Traffic assignment (Section 4.4.4).

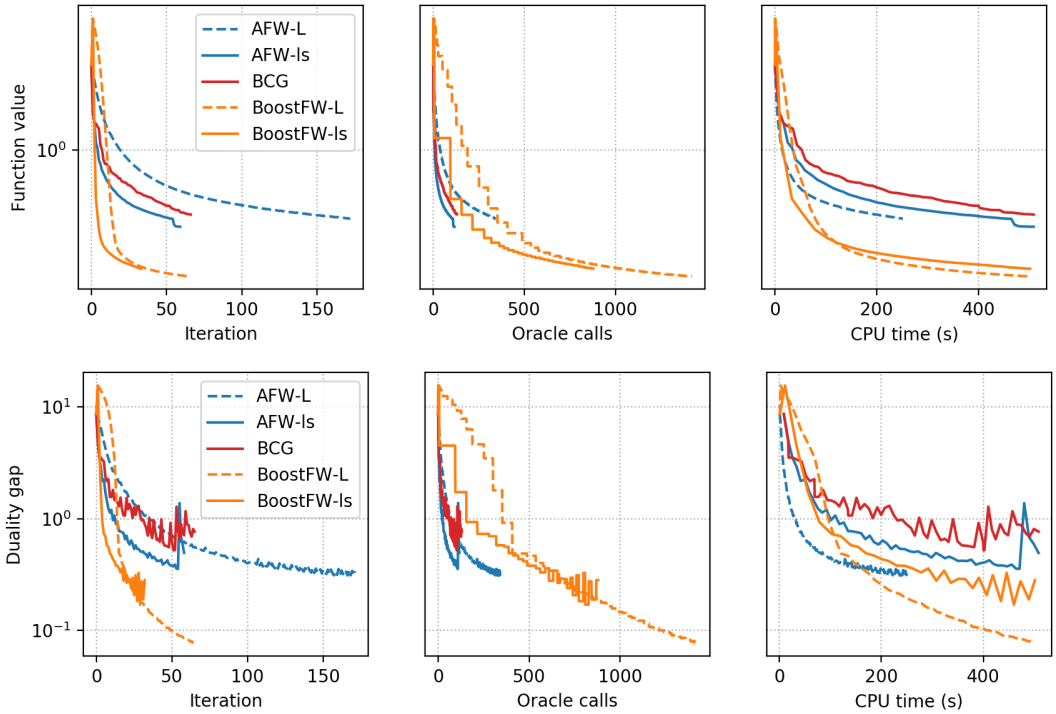

Figure B.4: Collaborative filtering on the MovieLens 100k dataset (Section 4.4.5).

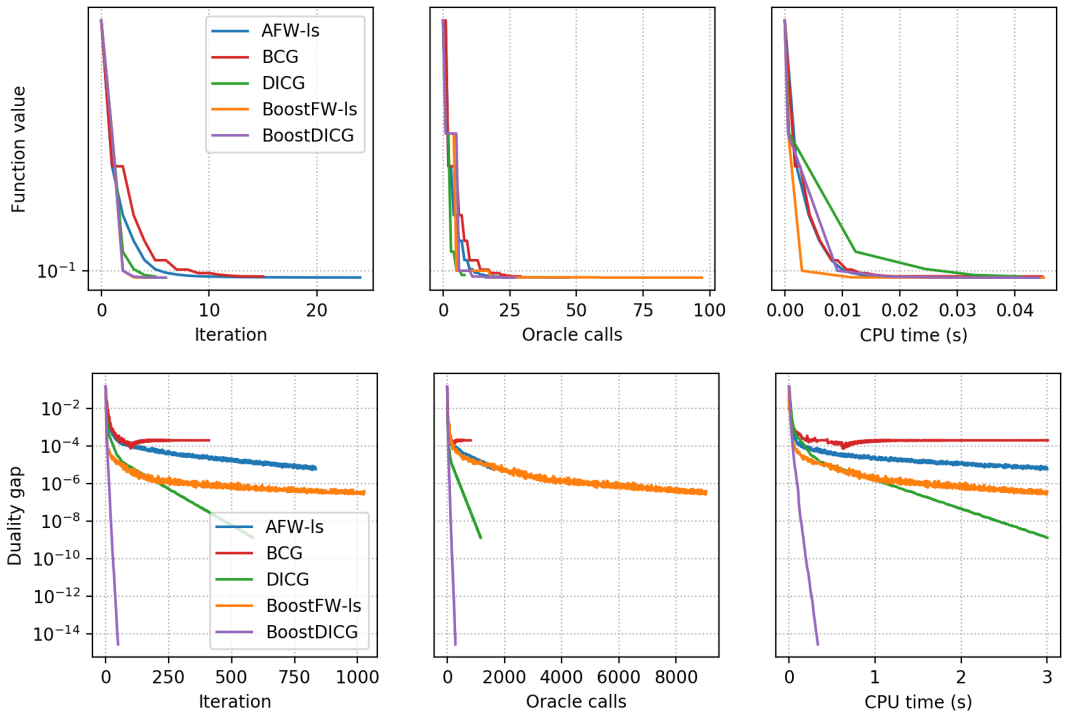

Figure B.5: Video co-localization on the YouTube-Objects dataset (Section 4.4.6).

B.3 Sensitivity analysis of AdaSFW

We report in Figures B.6–B.10 the sensitivity to K in the respective computational experiments (Section 5.5). In most cases, we can see that for large values of K, the method becomes less efficient in CPU time. This validates our approach, detailed in Section 5.4.1, since $K \gg 1$ represents solving the subproblems (almost) completely.

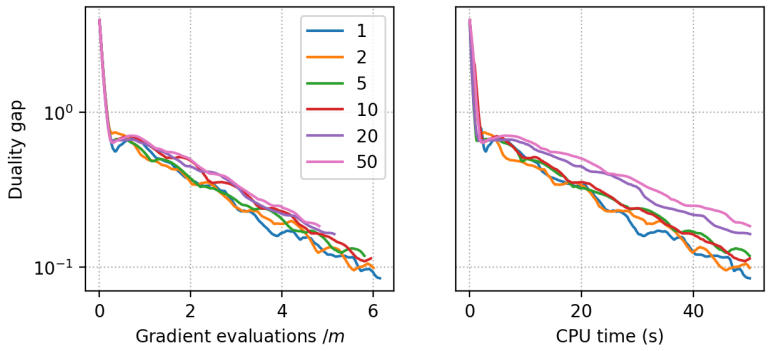

Figure B.6: Sensitivity of AdaCSFW to K on the support vector classification experiment.

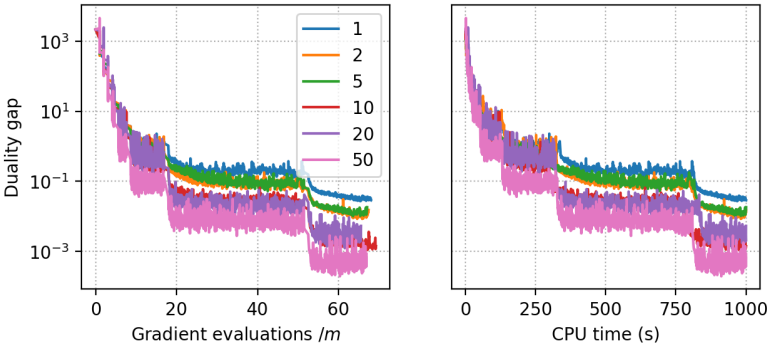

Figure B.7: Sensitivity of AdaSVRF to K on the linear regression experiment.

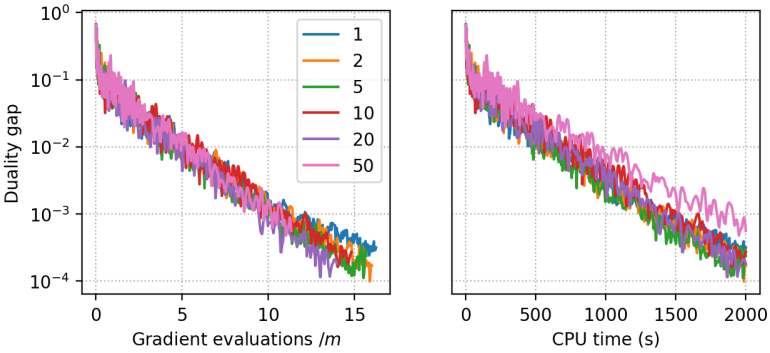

Figure B.8: Sensitivity of AdaCSFW to K on the logistic regression experiment.

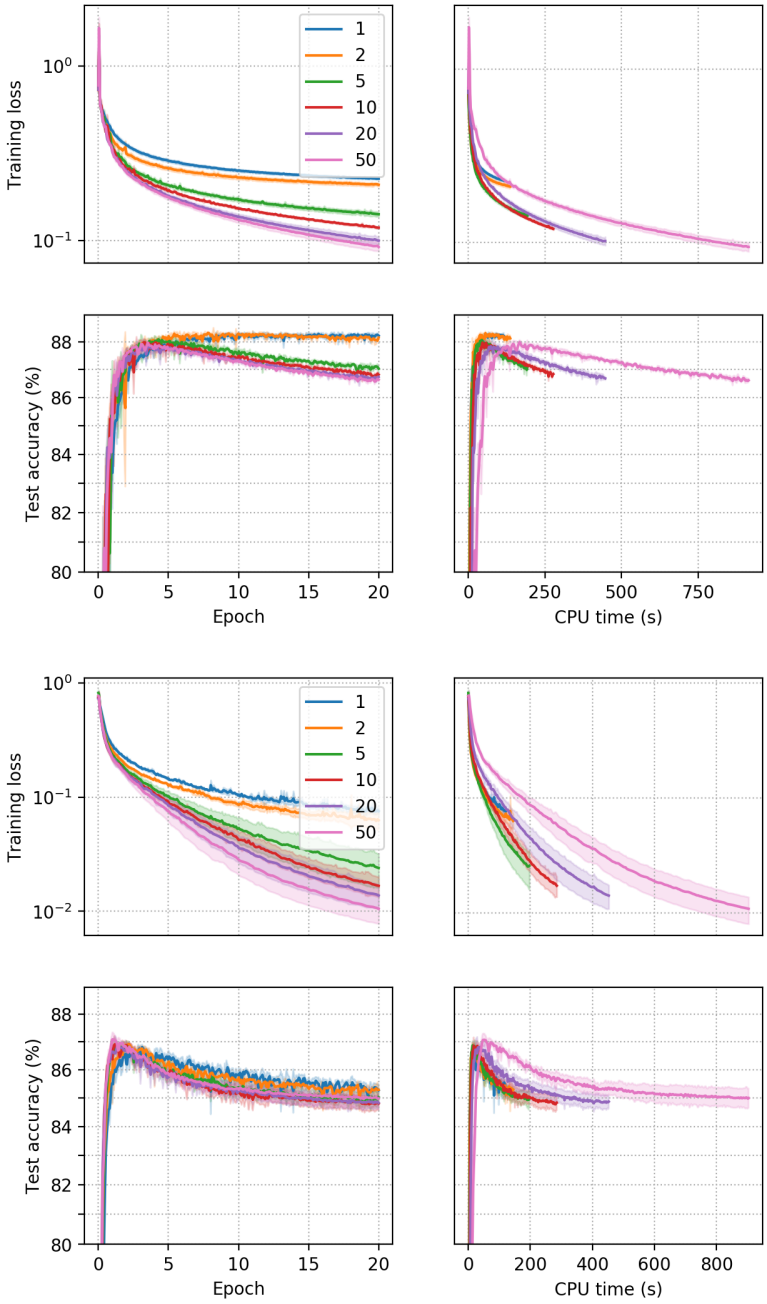

Figure B.9: Sensitivity of AdaSFW (top) and AdamSFW (bottom) to K on the IMDB dataset experiment.

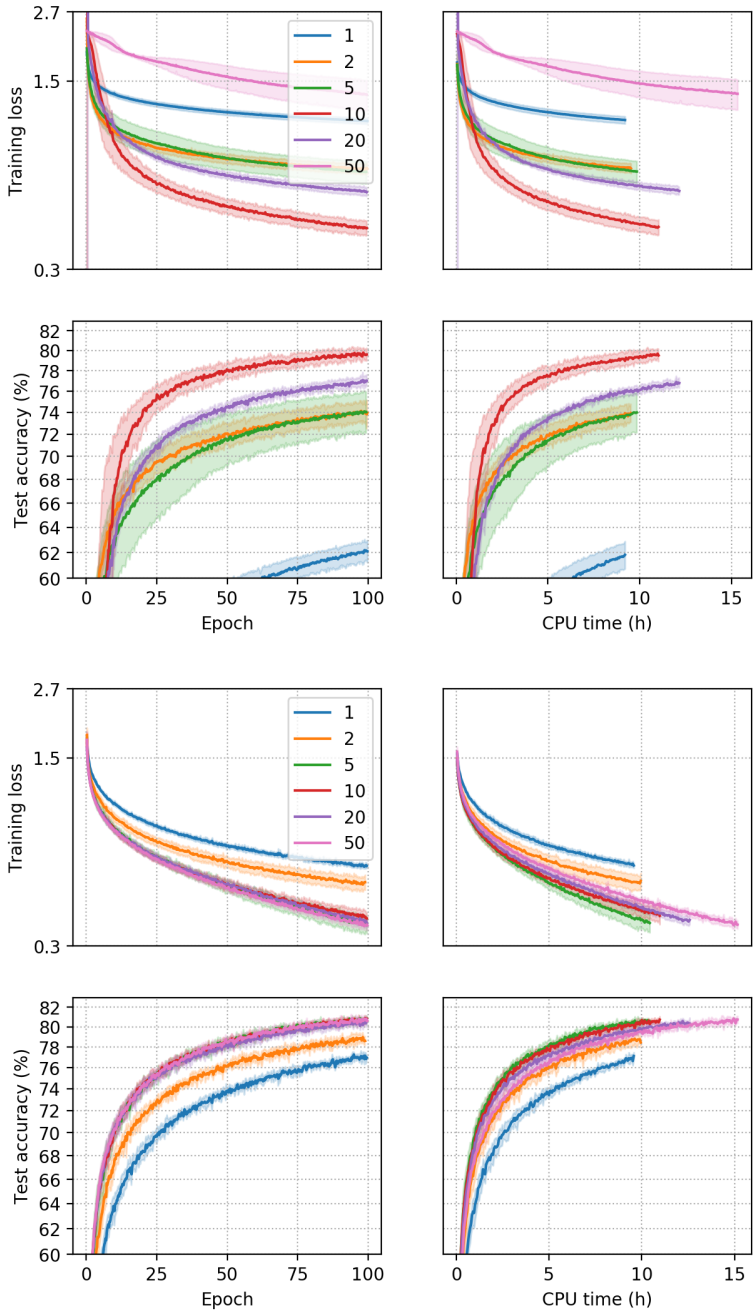

Figure B.10: Sensitivity of AdaSFW (top) and AdamSFW (bottom) to K on the CIFAR-10 dataset experiment.

B.4 Additional computational experiments for BMP

We provide the sensitivity analysis to the experiment in Figure 7.1 in Appendix B.4.1, and the comparison to the projected gradient method in Appendix B.4.2. We then con-

duct additional experiments on a variety of objective functions: an arbitrarily chosen norm (Appendix B.4.3), the Huber loss (Appendix B.4.4), the distance to a convex set (Appendix B.4.5), and a logistic regression loss (Appendix B.4.6).

B.4.1 Sensitivity of BMP to the parameter η

Here we report the sensitivity analysis of BMP for the data in Section 7.4.1. We ran BMP (Algorithm 7.3) for values of η in $\{100, 10, 5, 2, 1\}$. We set $\kappa = 2$ and $\tau = 2$ and did not activate the correction of atoms (Line 22). We report the results in Figure B.11.

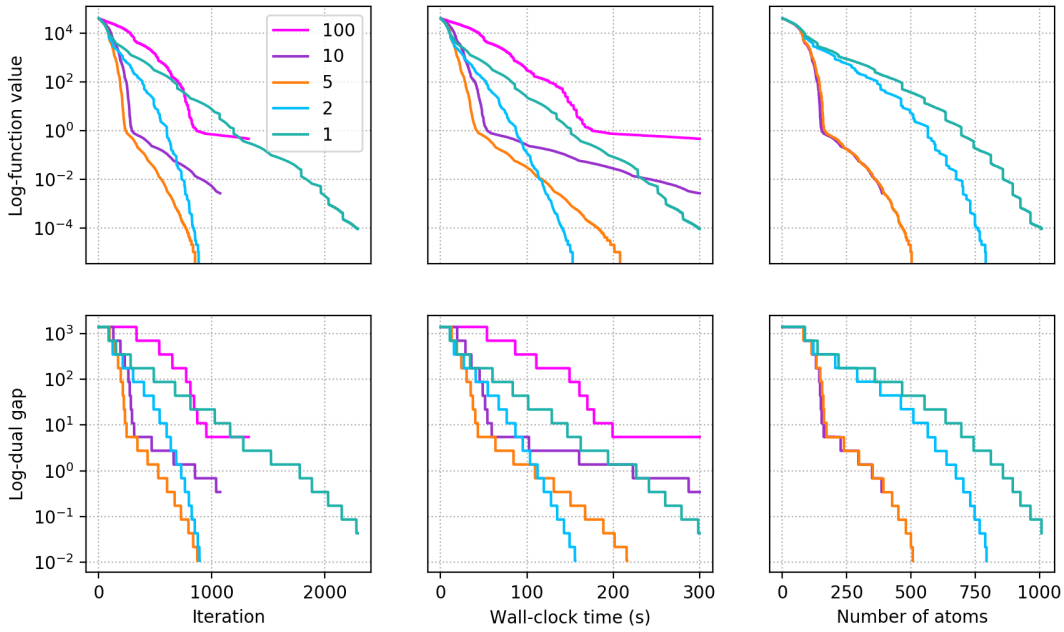

Figure B.11: Sensitivity of BMP to the parameter η.

We see that $\eta = 5$ is at the sweet spot between speed of convergence and sparsity of the iterates. Higher values of η have similar levels of sparsity but they perform worse for speed of convergence. Lower values of η perform much worse in sparsity and are not better in speed; $\eta = 2$ offsets $\eta = 5$ after 100 seconds but the function value is already 10^{-2} at that point. Therefore, by setting $\eta = 5$ in this example we achieve both speed of convergence and sparsity of the iterates. Similar insights are obtained in the other experiments, so that $\eta \sim 5$ seems to be a good initial choice.

For completeness, we present in Figure B.12 the sensitivity of BMP to η in NMSE.

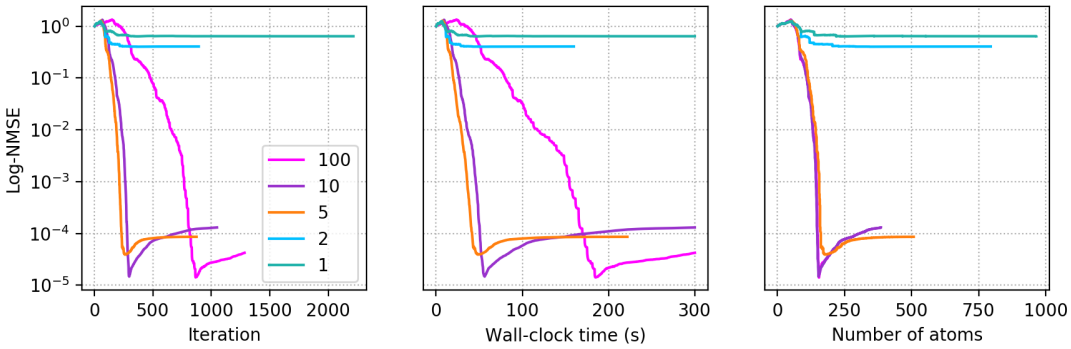

Figure B.12: Sensitivity of BMP to the parameter η in NMSE.

B.4.2 Comparison with PGD

Projected gradient descent (PGD) is a natural candidate for the experiment in Section 7.4.1. However, it does not ensure sufficient sparsity of the iterates. We depict three configurations of PGD in Figure B.13, each named "PGD:α" where PGD is ran with the constraint $\|x\|_1 \leqslant \alpha\|x^*\|_1$ and $\alpha \in \{1/2, 1, 2\}$. The implementation of PGD is in line with our general code framework and we used the method of [28] for projections onto the ℓ_1-ball. The number of atoms collected by the iterates in PGD are reported as the number of nonzero coordinates. Note that in BMP, GMP, and OMP we do not check if a selected atom v_t already satisfies $-v_t \in \mathcal{S}_t$ before adding it to $\mathcal{S}_t$, which is disadvantageous to these algorithms when evaluating their sparsity performance.

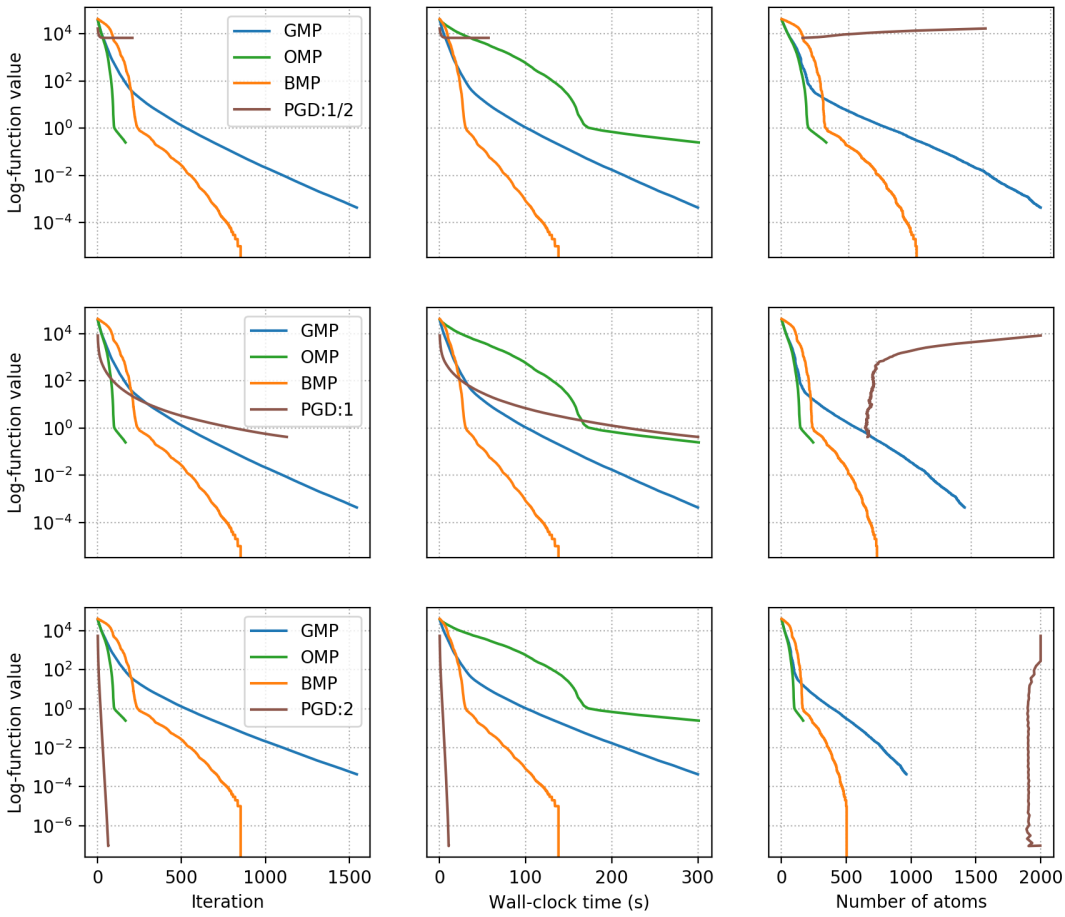

Figure B.13: Comparison of PGD vs. the MP algorithms.

As expected, the constraint $\|x\|_1 \leqslant \|x^*\|_1$ provides the best results for PGD; the constraint $\|x\|_1 \leqslant 2\|x^*\|_1$ is too loose and basically produces no sparsity in the iterates (recall that the ambient space is $\mathbb{R}^{2\,000}$). In the configuration $\|x\|_1 \leqslant \|x^*\|_1$, PGD does not converge faster than OMP and produces significantly worse sparsity than OMP and BMP.

B.4.3 Regression with arbitrarily chosen norm

We set $m = 250$, $n = 1\,000$, $s = 50$, and $\sigma = 0.05$ and generated the data as in Section 7.4.1. We ran a comparison with the arbitrarily chosen $f : x \in \mathbb{R}^n \mapsto \|Ax - b\|_3^5$. We plot the results in Figure B.14.

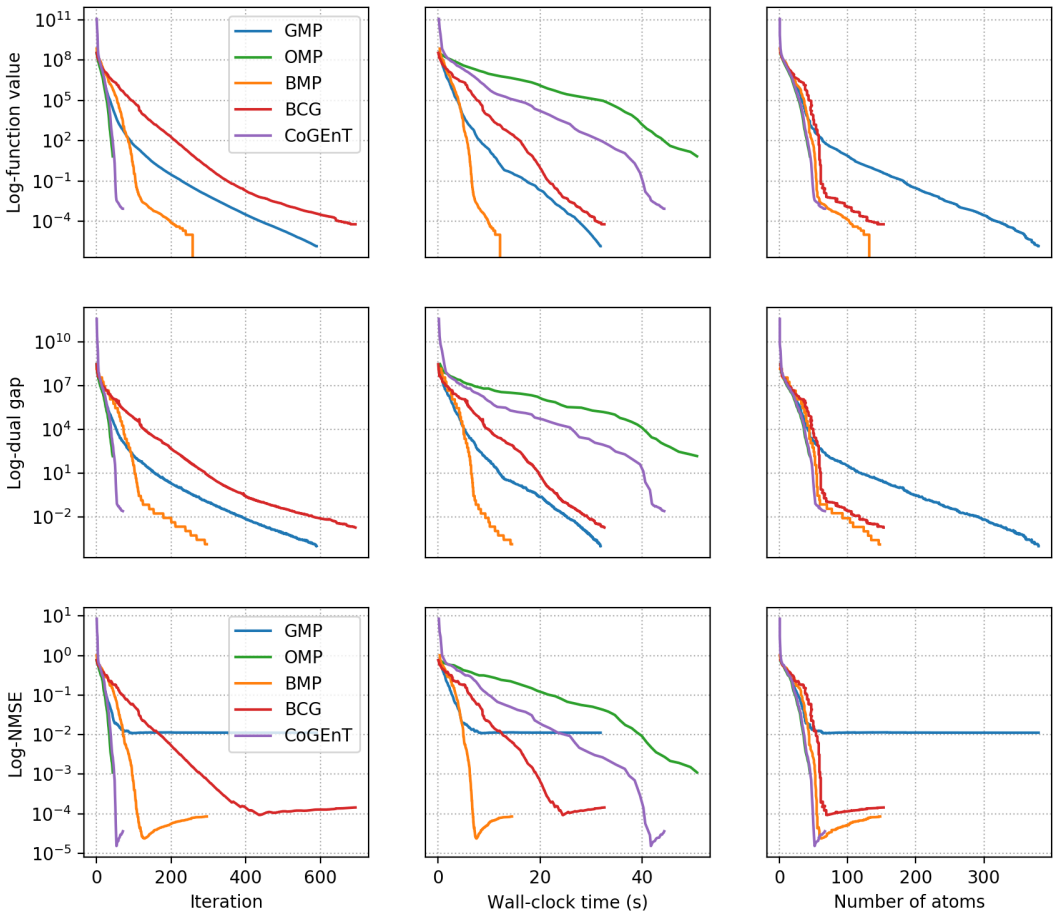

Figure B.14: Comparison of BMP vs. GMP and OMP on $f : x \in \mathbb{R}^n \mapsto \|Ax - b\|_3^5$, with $\eta = 5$.

We see that BMP offers very close-to-optimal levels of sparsity while being faster than the other algorithms in wall-clock time. We provide a sensitivity analysis of BMP to the parameter η in Figure B.15. The scaling is not exactly the same as in Figure B.14 due to the randomness in the generation of the data. We see that $\eta \sim 5$ is an appropriate choice combining the best of speed and sparsity.

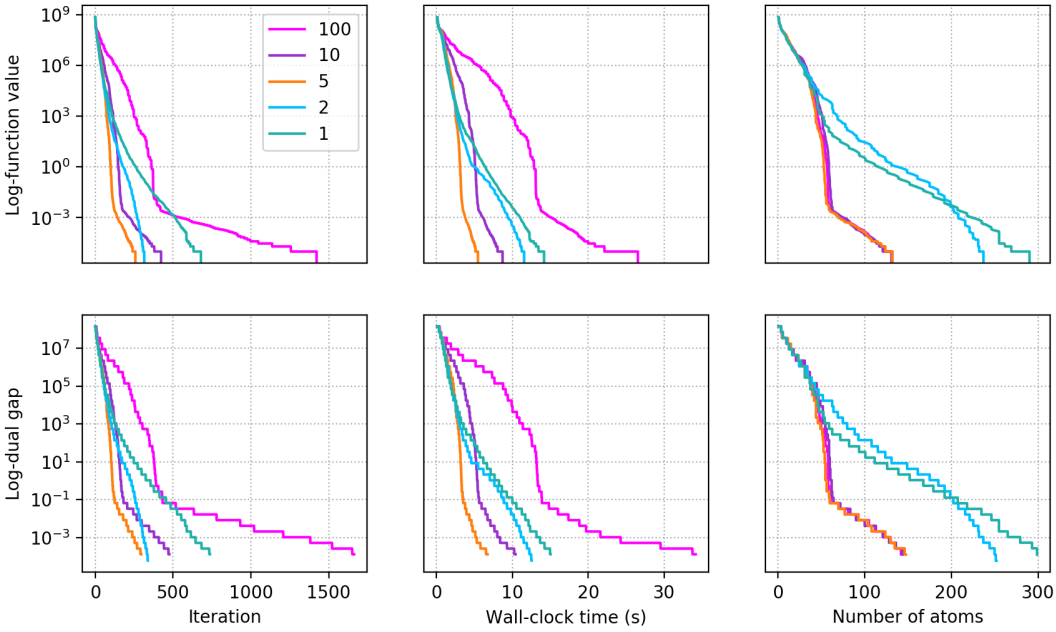

Figure B.15: Sensitivity of BMP to the parameter η.

For completeness, we present in Figure B.16 the sensitivity of BMP to η in NMSE.

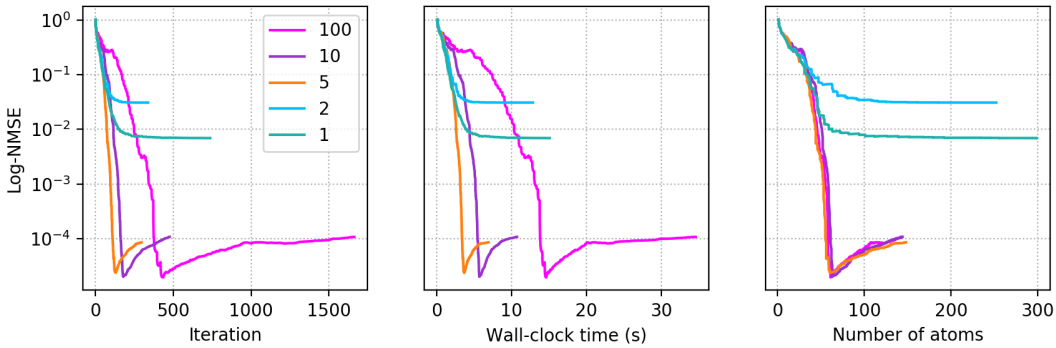

Figure B.16: Sensitivity of BMP to the parameter η in NMSE.

B.4.4 Huber loss

The Huber loss [61] is a smooth combination of the squared and absolute losses. The absolute loss is robust to outliers in the dataset, however its gradient is piecewise constant and not defined at the origin. This leads to instability of the solutions. The Huber loss

overcomes this by behaving like the squared loss around the origin:

$$h_\delta : t \in \mathbb{R} \mapsto \begin{cases} t^2/2 & \text{if } |t| \leqslant \delta \\ \delta(|t| - \delta/2) & \text{else} \end{cases}$$

where $\delta > 0$ defines this region around the origin. Note that the Huber loss is not strongly convex, as it is affine for $t > \delta$, however it is sharp as strong convexity holds around the origin.

We set $m = 250$, $n = 1\,000$, $s = 50$, and $\sigma = 0.05$ and generated the data as in Section 7.4.1. In this experiment we aim at minimizing the smooth convex function

$$f \colon x \in \mathbb{R}^n \mapsto \sum_{i=1}^{m} h_{10}(a_i^\top x - y_i)$$

where $a_1^\top, \ldots, a_m^\top \in \mathbb{R}^{1 \times n}$ are the rows of A. We plot the results in Figure B.17.

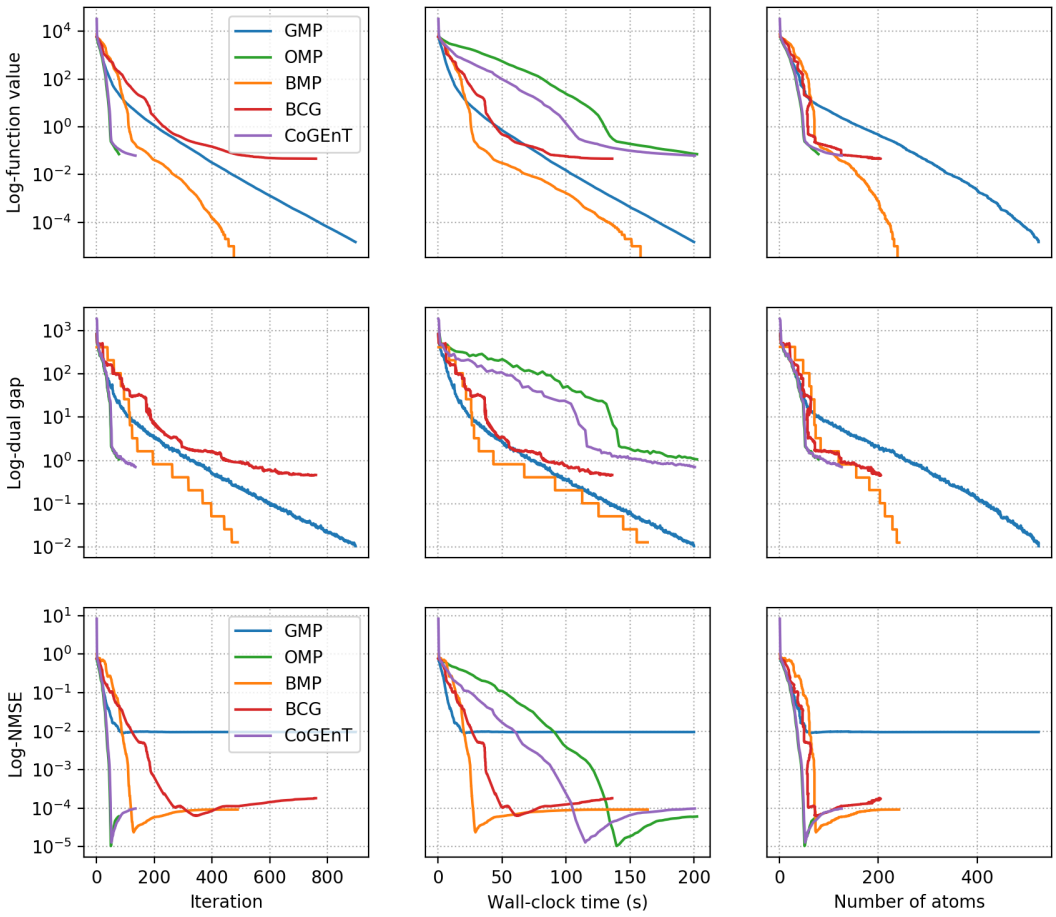

Figure B.17: Comparison of BMP vs. GMP and OMP on $f\colon x \in \mathbb{R}^n \mapsto \sum_{i=1}^m h_{10}(a_i^\top x - y_i)$, with $\eta = 5$.

Again, BMP has very close-to-optimal levels of sparsity while being the fastest algorithm to converge. We provide a sensitivity analysis of BMP to the parameter η in Figure B.18. The scaling is not exactly the same as in Figure B.17 due to the randomness in the generation of the data. We see that $\eta \sim 5$ is an appropriate choice combining the best of speed and sparsity.

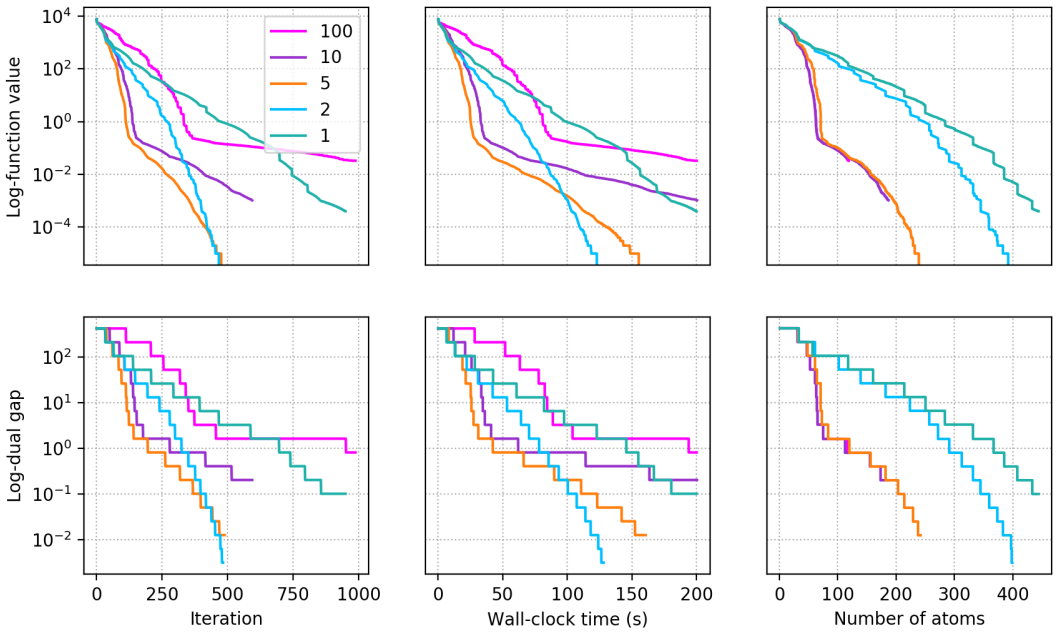

Figure B.18: Sensitivity of BMP to the parameter η.

For completeness, we present in Figure B.19 the sensitivity of BMP to η in NMSE.

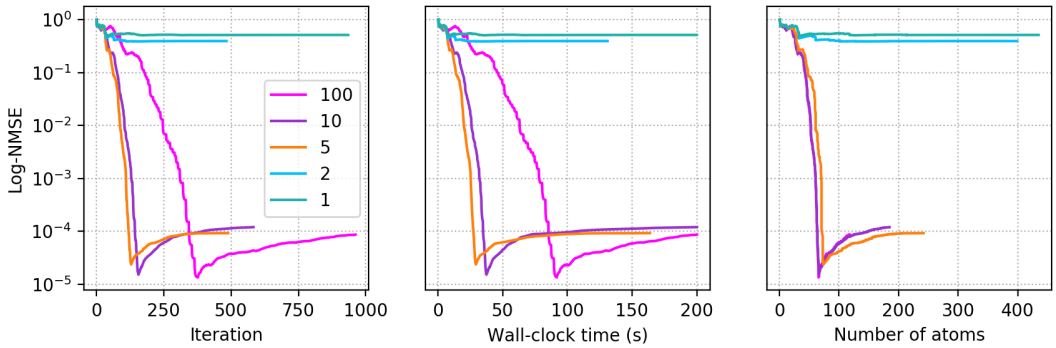

Figure B.19: Sensitivity of BMP to the parameter η in NMSE.

B.4.5 Distance to a convex set

Here we compared BMP vs. GMP and OMP on an arbitrarily chosen problem. We used

$$f\colon x \in \mathbb{R}^{500} \mapsto \operatorname{dist}(Ax-b, \overline{\mathcal{B}}(0,1))^2 = \|(Ax-b) - \operatorname{proj}(Ax-b, \overline{\mathcal{B}}(0,1))\|_2^2$$

and $\mathcal{D}$ a dictionary of 750 atoms randomly chosen in $\mathbb{R}^{500}$, where $A \in \mathbb{R}^{500\times 500}$ and $b \in \mathbb{R}^{500}$ are also randomly chosen. We did not reduce f to a closed-form expression simplifying computations. This is not a setting where BCG or CoGEnT can be applied. We depict two configurations of BMP: one with emphasis on sparsity of the iterates and one with emphasis on speed of convergence. The parameters η_{sparse} and η_{fast} were both optimized. We plot the results in Figure B.20.

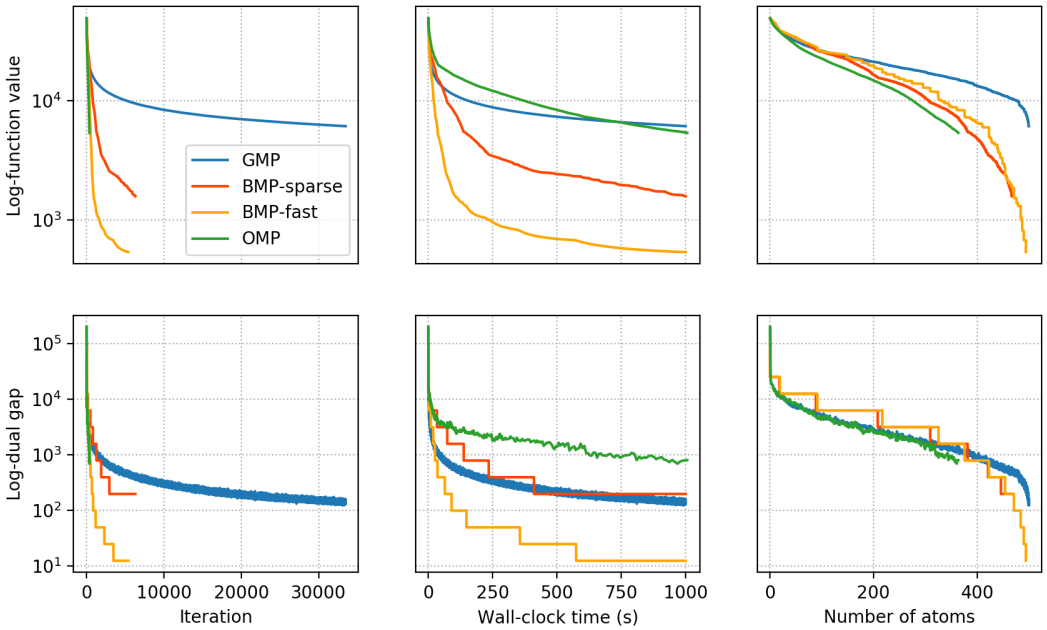

Figure B.20: Comparison of BMP, GMP, and OMP on $f\colon x \in \mathbb{R}^{500} \mapsto d(Ax-b, \overline{\mathcal{B}}(0,1))^2$, with $\eta_{\text{sparse}} = 10$ and $\eta_{\text{fast}} = 2$.

We provide a sensitivity analysis of BMP to the parameter η in Figure B.21. The scaling is not exactly the same as in Figure B.20 due to the randomness in the generation of the data. We see that $\eta \sim 2$ is an appropriate choice combining the best of speed and sparsity.

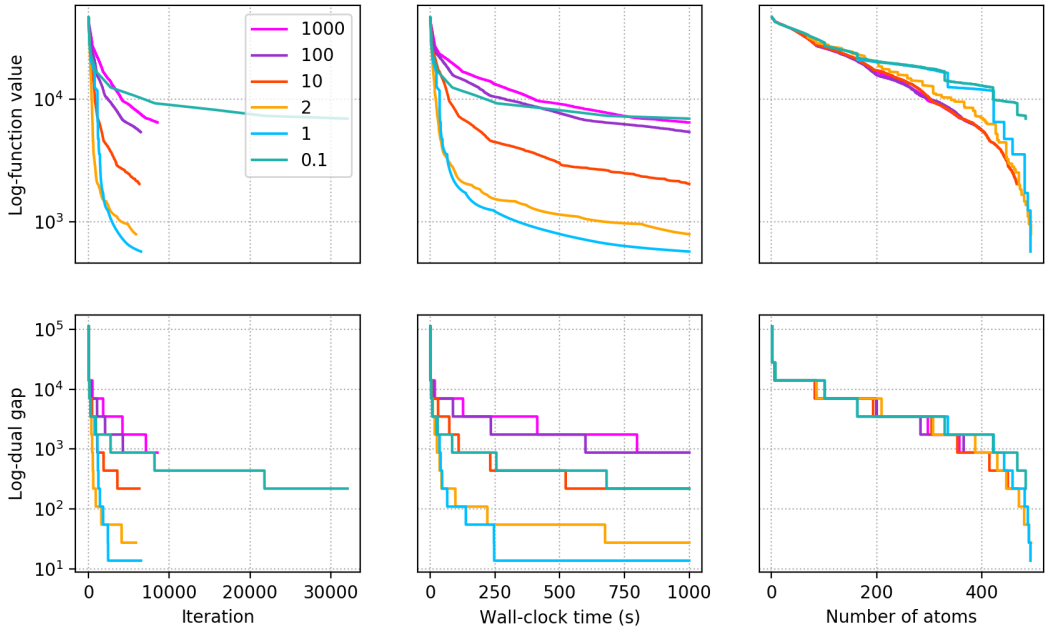

Figure B.21: Sensitivity of BMP to the parameter η.

B.4.6 Logistic regression

Here we compared BMP, GMP, and OMP on the Gisette dataset [51] available at https://archive.ics.uci.edu/ml/datasets/Gisette. We did not have access to the true parameters x^* so we could not produce the NMSE plots. The objective function is the logistic loss

$$f\colon x \in \mathbb{R}^n \mapsto \frac{1}{m}\sum_{i=1}^{m} \ln(1 + e^{-y_i a_i^\top x})$$

with the labels $y_i \in \{-1, 1\}$, and the dictionary is the set of signed canonical vectors $\mathcal{D} = \{\pm e_1, \dots, e_n\}$. We have $n = 5\,000$ and in order to reduce the running time, we chose $m = 1\,000$ and we slightly enhanced the code framework by replacing $\min_{v\in\mathcal{D}} \langle v, \nabla f(x_t)\rangle$ with $\min_{i\in[\![1,n]\!]} -|\,[\nabla f(x_t)]_i\,|$. Note that this lessens the speed-up provided by the weak-separation oracle in BMP. We represented the duality gaps of BMP by $\max_{i\in[\![1,n]\!]} |\,[\nabla f(x_t)]_i\,|$, like for GMP and OMP, and thus yielding a similar zig-zagging plot.

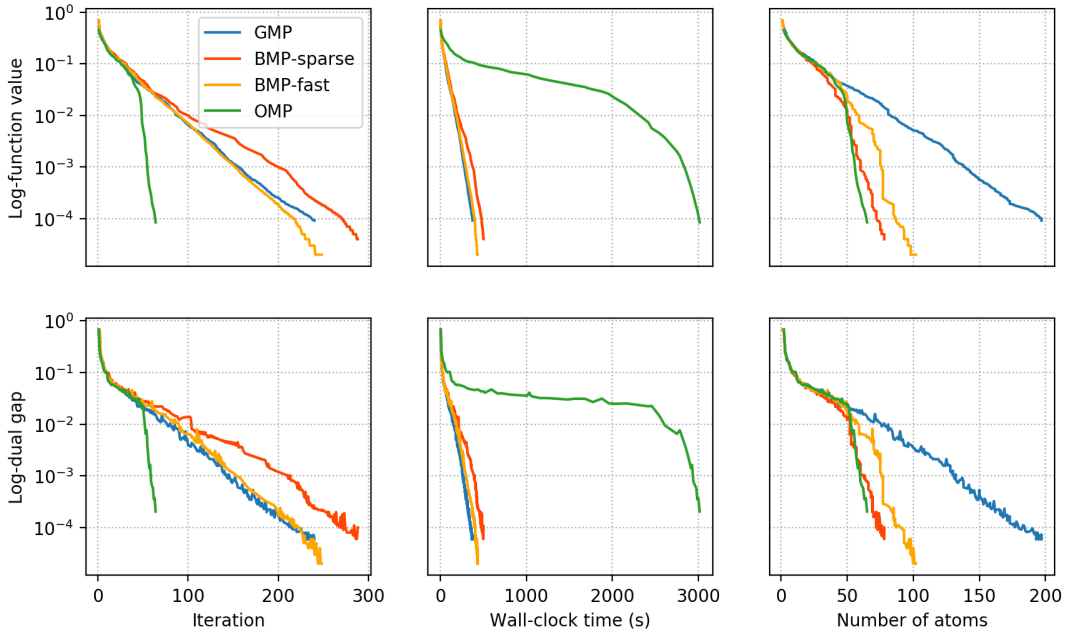

Figure B.22: Comparison of BMP, GMP, and OMP on the Gisette dataset with $\eta_{\text{sparse}} = 3$ and $\eta_{\text{fast}} = 2$.

In this situation, Figure B.22 shows that BMP converges as fast as GMP while producing iterates with much higher sparsity, equivalent to that of OMP. Hence, BMP hits the sweet spot of speed of convergence and sparsity of the iterates.

REFERENCES

[1] A. Argyriou, M. Signoretto, and J. A. K. Suykens, "Hybrid conditional gradient-smoothing algorithms with applications to sparse and low rank regularization," in *Regularization, Optimization, Kernels, and Support Vector Machines*, Chapman & Hall/CRC, 2014, pp. 53–82.

[2] F. Bach, "Duality between subgradient and conditional gradient methods," *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 115–129, 2015.

[3] S. Barman, "Approximating Nash equilibria and dense bipartite subgraphs via an approximate version of Carathéodory's theorem," in *Proceedings of the 47th Annual ACM Symposium on Theory of Computing*, 2015, pp. 361–369.

[4] M. A. Bashiri and X. Zhang, "Decomposition-invariant conditional gradient for general polytopes with line search," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 2690–2700.

[5] H. H. Bauschke and J. M. Borwein, "On projection algorithms for solving convex feasibility problems," *SIAM Review*, vol. 38, no. 3, pp. 367–426, 1996.

[6] H. H. Bauschke and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*, 2nd. Springer, 2017.

[7] M. J. Beckmann, C. B. McGuire, and C. B. Winsten, *Studies in the Economics of Transportation*. Yale University Press, 1956.

[8] A. Ben-Tal and A. S. Nemirovski, *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Society for Industrial and Applied Mathematics, 2001.

[9] T. Bertin-Mahieux, D. P. W. Ellis, B. Whitman, and P. Lamere, "The Million Song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval*, 2011.

[10] L. Biewald, *Experiment tracking with weights and biases*, Software available at https://www.wandb.com, 2020.

[11] J. Bolte, A. Daniilidis, and A. Lewis, "The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems," *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 1205–1223, 2007.

[12] J. Bolte, T.-P. Nguyen, J. Peypouquet, and B. W. Suter, "From error bounds to the complexity of first-order descent methods for convex functions," *Mathematical Programming*, vol. 165, no. 2, pp. 471–507, 2017.

[13] J. Bourgain, A. Pajor, S. J. Szarek, and N. Tomczak-Jaegermann, "On the duality problem for entropy numbers of operators," in *Geometric Aspects of Functional Analysis*, Springer, 1989, pp. 50–63.

[14] G. Braun, S. Pokutta, D. Tu, and S. Wright, "Blended conditional gradients: The unconditioning of conditional gradients," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 735–743.

[15] G. Braun, S. Pokutta, and D. Zink, "Lazifying conditional gradient algorithms," *Journal of Machine Learning Research*, vol. 20, no. 71, pp. 1–42, 2019.

[16] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[17] M. D. Canon and C. D. Cullum, "A tight upper bound on the rate of convergence of Frank-Wolfe algorithm," *SIAM Journal on Control*, vol. 6, no. 4, pp. 509–516, 1968.

[18] C. Carathéodory, "Über den Variabilitätsbereich der Koeffizienten von Potenzreihen, die gegebene Werte nicht annehmen," *Mathematische Annalen*, vol. 64, no. 1, pp. 95–115, 1907.

[19] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, 2011.

[20] J. Chen, D. Zhou, J. Yi, and Q. Gu, "A Frank-Wolfe framework for efficient and effective adversarial attacks," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 2020, pp. 3486–3494.

[21] K. L. Clarkson, "Coresets, sparse greedy approximation, and the Frank-Wolfe algorithm," *ACM Transactions on Algorithms*, vol. 6, no. 4, pp. 1–30, 2010.

[22] C. W. Combettes and S. Pokutta, "Blended matching pursuit," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 2044–2054.

[23] ——, "Boosting Frank-Wolfe by chasing gradients," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, pp. 2111–2121.

[24] ——, "Complexity of linear minimization and projection on some sets," *arXiv preprint arXiv:2101.10040*, 2021.

[25] ——, "Revisiting the approximate Carathéodory problem via the Frank-Wolfe algorithm," *arXiv preprint arXiv:1911.04415*, 2019.

[26] C. W. Combettes, C. Spiegel, and S. Pokutta, "Projection-free adaptive gradients for large-scale optimization," *arXiv preprint arXiv:2009.14114*, 2020.

[27] P. L. Combettes, "Strong convergence of block-iterative outer approximation methods for convex optimization," *SIAM Journal on Control and Optimization*, vol. 38, no. 2, pp. 538–565, 2000.

[28] L. Condat, "Fast projection onto the simplex and the ℓ_1 ball," *Mathematical Programming*, vol. 158, no. 1, pp. 575–585, 2016.

[29] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd. MIT Press, 2009.

[30] A. Cutkosky and F. Orabona, "Momentum-based variance reduction in non-convex SGD," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 15 236–15 245.

[31] W. Dai and O. Milenkovic, "Subspace pursuit for compressive sensing signal reconstruction," *IEEE Transactions on Information Theory*, vol. 55, no. 5, pp. 2230–2249, 2009.

[32] D. Davis and W. Yin, "Faster convergence rates of relaxed Peaceman-Rachford and ADMM under regularity assumptions," *Mathematics of Operations Research*, vol. 42, no. 3, pp. 783–805, 2017.

[33] G. Davis, S. Mallat, and Z. Zhang, "Adaptive time-frequency decompositions with matching pursuits," *Optical Engineering*, vol. 33, no. 7, pp. 2183–2191, 1994.

[34] J. Dean et al, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, vol. 25, 2012, pp. 1223–1231.

[35] A. Defazio and L. Bottou, "On the ineffectiveness of variance reduced optimization for deep learning," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 1755–1765.

[36] V. F. Demyanov and A. M. Rubinov, *Approximate Methods in Optimization Problems*. American Elsevier, 1970.

[37] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 61, pp. 2121–2159, 2011.

[38] J. C. Duchi, "Introductory lectures on stochastic optimization," in *The Mathematics of Data*, American Mathematical Society, 2018.

[39] J. C. Dunn and S. Harshbarger, "Conditional gradient algorithms with open loop step size rules," *Journal of Mathematical Analysis and Applications*, vol. 62, no. 2, pp. 432–444, 1978.

[40] I. Ekeland and R. Témam, *Convex Analysis and Variational Problems*. Society for Industrial and Applied Mathematics, 1999, Originally published in 1976.

[41] C. Fang, C. J. Li, Z. Lin, and T. Zhang, "SPIDER: Near-optimal non-convex optimization via stochastic path-integrated differential estimator," in *Advances in Neural Information Processing Systems*, vol. 31, 2018, pp. 689–699.

[42] M. Fazel, H. Hindi, and S. P. Boyd, "A rank minimization heuristic with application to minimum order system approximation," in *Proceedings of the 2001 American Control Conference*, 2001, pp. 4734–4739.

[43] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, no. 1–2, pp. 95–110, 1956.

[44] D. Garber, "Projection-free algorithms for convex optimization and online learning," Ph.D. thesis, Technion, 2016.

[45] D. Garber and E. Hazan, "Faster rates for the Frank-Wolfe method over strongly-convex sets," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015, pp. 541–549.

[46] D. Garber and O. Meshi, "Linear-memory and decomposition-invariant linearly convergent conditional gradient algorithm for structured polytopes," in *Advances in Neural Information Processing Systems*, vol. 29, 2016, pp. 1001–1009.

[47] D. Garber and N. Wolf, "Frank-Wolfe with a nearest extreme point oracle," *arXiv preprint arXiv:2102.02029*, 2021.

[48] R. Garg and R. Khandekar, "Gradient descent with sparsification: An iterative algorithm for sparse recovery with restricted isometry property," in *Proceedings of the 26th International Conference on Machine Learning*, 2009, pp. 337–344.

[49] G. H. Golub and C. F. van Loan, *Matrix Computations*, 4th. Johns Hopkins University Press, 2013.

[50] J. Guélat and P. Marcotte, "Some comments on Wolfe's 'away step'," *Mathematical Programming*, vol. 35, no. 1, pp. 110–119, 1986.

[51] I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror, "Result analysis of the NIPS 2003 feature selection challenge," in *Advances in Neural Information Processing Systems*, vol. 17, 2005, pp. 545–552.

[52] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proceedings of the 7th Python in Science Conference*, 2008, pp. 11–15.

[53] F. M. Harper and J. A. Konstan, "The MovieLens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, 19:1–19:19, 2015.

[54] C. R. Harris et al, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[55] Y. Haugazeau, "Sur les inéquations variationnelles et la minimisation de fonctionnelles convexes," Thèse de doctorat, Université de Paris, 1968.

[56] E. Hazan, "Sparse approximate solutions to semidefinite programs," in *Proceedings of the 8th Latin American Symposium on Theoretical Informatics*, 2008, pp. 306–316.

[57] E. Hazan and S. Kale, "Projection-free online learning," in *Proceedings of the 29th International Conference on Machine Learning*, 2012.

[58] E. Hazan and H. Luo, "Variance-reduced and projection-free stochastic optimization," in *Proceedings of the 33rd International Conference on Machine Learning*, 2016, pp. 1263–1271.

[59] D. W. Hearn, "The gap function of a convex program," *Operations Research Letters*, vol. 1, no. 2, pp. 67–71, 1982.

[60] C. A. Holloway, "An extension of the Frank and Wolfe method of feasible directions," *Mathematical Programming*, vol. 6, no. 1, pp. 14–27, 1974.

[61] P. J. Huber, "Robust estimation of a location parameter," *The Annals of Mathematical Statistics*, vol. 35, no. 1, pp. 73–101, 1964.

[62] G. Ivanov, "Approximate Carathéodory's theorem in uniformly smooth Banach spaces," *Discrete and Computational Geometry*, 2019.

[63] M. Jaggi, "Revisiting Frank-Wolfe: Projection-free sparse convex optimization," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 427–435.

[64] M. Jaggi and M. Sulovský, "A simple algorithm for nuclear norm regularized problems," in *Proceedings of the 27th International Conference on Machine Learning*, 2010, pp. 471–478.

[65] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in Neural Information Processing Systems*, vol. 26, 2013, pp. 315–323.

[66] A. Joulin, K. Tang, and L. Fei-Fei, "Efficient image and video co-localization with Frank-Wolfe algorithm," in *European Conference on Computer Vision*, 2014, pp. 253–268.

[67] T. Kerdreux, A. d'Aspremont, and S. Pokutta, "Projection-free optimization on uniformly convex sets," in *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, 2021, pp. 19–27.

[68] ——, "Restarting Frank-Wolfe," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, 2019, pp. 1275–1283.

[69] T. Kerdreux, F. Pedregosa, and A. d'Aspremont, "Frank-Wolfe with subsampling oracle," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 2591–2600.

[70] N. S. Keskar and R. Socher, "Improving generalization performance by switching from Adam to SGD," *arXiv preprint arXiv:1712.07628*, 2017.

[71] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, 2015.

[72] A. V. Knyazev, M. E. Argentati, I. Lashuk, and E. E. Ovtchinnikov, "Block locally optimal preconditioned eigenvalue xolvers (BLOPEX) in hypre and PETSc," *SIAM Journal on Scientific Computing*, vol. 29, no. 5, pp. 2224–2239, 2007.

[73] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, 2009.

[74] J. Kuczyński and H. Woźniakowski, "Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start," *SIAM Journal on Matrix Analysis and Applications*, vol. 13, no. 4, pp. 1094–1122, 1992.

[75] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1–2, pp. 83–97, 1955.

[76] K. Kurdyka, "On gradients of functions definable in o-minimal structures," *Annales de l'Institut Fourier*, vol. 48, no. 3, pp. 769–783, 1998.

[77] S. Lacoste-Julien, "Convergence rate of Frank-Wolfe for non-convex objectives," *arXiv preprint arXiv:1607.00345*, 2016.

[78] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of Frank-Wolfe optimization variants," in *Advances in Neural Information Processing Systems*, vol. 28, 2015, pp. 496–504.

[79] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher, "Block-coordinate Frank-Wolfe optimization for structural SVMs," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 53–61.

[80] G. Lan, "The complexity of large-scale convex programming under a linear optimization oracle," Department of Industrial and Systems Engineering, University of Florida, Tech. Rep., 2013.

[81] G. Lan, S. Pokutta, Y. Zhou, and D. Zink, "Conditional accelerated lazy stochastic gradient descent," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1965–1974.

[82] G. Lan and Y. Zhou, "Conditional gradient sliding for convex optimization," *SIAM Journal on Optimization*, vol. 26, no. 2, pp. 1379–1409, 2016.

[83] L. J. LeBlanc, E. K. Morlok, and W. P. Pierskalla, "An efficient approach to solving the road network equilibrium traffic assignment problem," *Transportation Research*, vol. 9, no. 5, pp. 309–318, 1975.

[84] R. B. Lehoucq, D. C. Sorensen, and C. Yang, *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*. Society for Industrial and Applied Mathematics, 1998.

[85] E. S. Levitin and B. T. Polyak, "Constrained minimization methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 6, no. 5, pp. 1–50, 1966.

[86] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "RCV1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[87] P.-L. Lions and B. Mercier, "Splitting algorithms for the sum of two nonlinear operators," *SIAM Journal on Numerical Analysis*, vol. 16, no. 6, pp. 964–979, 1979.

[88] F. Locatello, R. Khanna, M. Tschannen, and M. Jaggi, "A unified optimization view on generalized matching pursuit and Frank-Wolfe," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017, pp. 860–868.

[89] F. Locatello, A. Raj, S. P. Karimireddy, G. Rätsch, B. Schölkopf, S. U. Stich, and M. Jaggi, "On matching pursuit and coordinate descent," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 3198–3207.

[90] F. Locatello, M. Tschannen, G. Rätsch, and M. Jaggi, "Greedy algorithms for cone constrained optimization with convergence guarantees," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 773–784.

[91] S. Łojasiewicz, "Une propriété topologique des sous-ensembles analytiques réels," in *Les Équations aux Dérivées Partielles*, ser. 117, Colloques Internationaux du CNRS, 1963, pp. 87–89.

[92] G. Luise, S. Salzo, M. Pontil, and C. Ciliberto, "Sinkhorn barycenters with free support via Frank-Wolfe algorithm," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 9322–9333.

[93] L. Luo, Y. Xiong, Y. Liu, and X. Sun, "Adaptive gradient methods with dynamic bound of learning rate," in *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[94] A. Maalouf, I. Jubran, and D. Feldman, "Fast and accurate least-mean-squares solvers," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 8307–8318.

[95] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, 2011, pp. 142–150.

[96] S. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3397–3415, 1993.

[97] H. B. McMahan and M. Streeter, "Adaptive bound optimization for online convex optimization," in *Proceedings of the 23rd Annual Conference on Learning Theory*, 2010.

[98] B. Mehta, T. Hofmann, and W. Nejdl, "Robust collaborative filtering," in *Proceedings of the 2007 ACM Conference on Recommender Systems*, 2007, pp. 49–56.

[99] H. Mine and M. Fukushima, "A minimization method for the sum of a convex function and a continuously differentiable function," *Journal of Optimization Theory and Applications*, vol. 33, pp. 9–23, 1981.

[100] V. Mirrokni, R. Paes Leme, A. Vladu, and S. C.-W. Wong, "Tight bounds for approximate Carathéodory and beyond," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 2440–2448.

[101] J. J. Moreau, "Fonctions convexes duales et points proximaux dans un espace hilbertien," *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences*, vol. 255, pp. 2897–2899, 1962.

[102] ——, "Proximité et dualité dans un espace hilbertien," *Bulletin de la Société Mathématique de France*, vol. 93, pp. 273–279, 1965.

[103] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[104] D. Needell and J. A. Tropp, "CoSaMP: Iterative signal recovery from incomplete and inaccurate samples," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 301–321, 2009.

[105] G. Négiar, G. Dresdner, A. Y.-T. Tsai, L. El Ghaoui, F. Locatello, R. M. Freund, and F. Pedregosa, "Stochastic Frank-Wolfe for constrained finite-sum minimization," in *Proceedings of the 37th International Conference on Machine Learning*, 2020, pp. 7253–7262.

[106] R. Negrinho and A. F. T. Martins, "Orbit regularization," in *Advances in Neural Information Processing Systems*, vol. 27, 2014, pp. 3221–3229.

[107] A. S. Nemirovskii and Y. E. Nesterov, "Optimal methods of smooth convex minimization," *USSR Computational Mathematics and Mathematical Physics*, vol. 25, no. 2, pp. 21–30, 1985.

[108] A. S. Nemirovsky and D. B. Yudin, *Problem Complexity and Method Efficiency in Optimization*. Wiley, 1983.

[109] C. H. J. Pang, "First order constrained optimization algorithms with feasibility updates," *arXiv preprint arXiv:1506.08247*, 2015.

[110] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of the 27th Asilomar Conference on Signals, Systems, and Computers*, 1993, pp. 40–44.

[111] F. Pedregosa, G. Négiar, A. Askari, and M. Jaggi, "Linearly convergent Frank-Wolfe with backtracking line-search," in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2020, pp. 1–10.

[112] R. Penrose, "A generalized inverse for matrices," *Mathematical Proceedings of the Cambridge Philosophical Society*, vol. 51, no. 3, pp. 406–413, 1955.

[113] G. Pisier, "Remarques sur un résultat non publié de B. Maurey," in *Séminaire d'Analyse Fonctionnelle*, ser. 5, École Polytechnique, 1981, pp. 1–12.

[114] B. T. Polyak, "Gradient methods for the minimisation of functionals," *USSR Computational Mathematics and Mathematical Physics*, vol. 3, no. 4, pp. 864–878, 1963.

[115] A. Prest, C. Leistner, J. Civera, C. Schmid, and V. Ferrari, "Learning object class detectors from weakly annotated video," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3282–3289.

[116] N. Rao, S. Shah, and S. Wright, "Forward-backward greedy algorithms for atomic norm regularization," *IEEE Transactions on Signal Processing*, vol. 63, no. 21, pp. 5798–5811, 2015.

[117] S. J. Reddi, S. Kale, and S. Kumar, "On the convergence of Adam and beyond," in *Proceedings of the 6th International Conference on Learning Representations*, 2018.

[118] S. J. Reddi, S. Sra, B. Póczos, and A. Smola, "Stochastic Frank-Wolfe methods for nonconvex optimization," in *54th Annual Allerton Conference on Communication, Control, and Computing*, 2016, pp. 1244–1251.

[119] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[120] V. Roulet and A. d'Aspremont, "Sharpness, restart and acceleration," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 1119–1129.

[121] M. Schmidt, N. Le Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," *Mathematical Programming*, vol. 162, no. 1–2, pp. 83–112, 2017.

[122] S. Shalev-Shwartz, "Online learning: Theory, algorithms, and applications," Ph.D. thesis, Hebrew University, 2007.

[123] Z. Shen, C. Fang, P. Zhao, J. Huang, and H. Qian, "Complexities in projection-free stochastic non-convex minimization," in *Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics*, 2019, pp. 2868–2876.

[124] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[125] T. Tieleman and G. Hinton, "Lecture 6e – rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, vol. 4, no. 2, pp. 26–31, 2012.

[126] V. Vapnik, "Principles of risk minimization for learning theory," in *Advances in Neural Information Processing Systems*, vol. 4, 1991, pp. 831–838.

[127] L. A. Végh, "A strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives," *SIAM Journal on Computing*, vol. 45, no. 5, pp. 1729–1761, 2016.

[128] P. Virtanen et al, "Scipy 1.0: Fundamental algorithms for scientific computing in Python," *Nature Methods*, vol. 17, no. 3, pp. 261–272, 2020.

[129] J. G. Wardrop, "Some theoretical aspects of road traffic research," in *Proceedings of the Institute of Civil Engineers*, vol. 1, 1952, pp. 325–378.

[130] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 4148–4158.

[131] P. Wolfe, "Convergence theory in nonlinear programming," in *Integer and Nonlinear Programming*, North-Holland, 1970, pp. 1–36.

[132] J. Xie, Z. Shen, C. Zhang, H. Qian, and B. Wang, "Efficient projection-free online methods with stochastic recursive gradient," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, 2020, pp. 6446–6453.

[133] X.-T. Yuan, P. Li, and T. Zhang, "Gradient hard thresholding pursuit," *Journal of Machine Learning Research*, vol. 18, no. 166, pp. 1–43, 2018.

[134] A. Yurtsever, S. Sra, and V. Cevher, "Conditional gradient methods via stochastic path-integrated differential estimator," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 7282–7291.

[135] C. Zălinescu, *Convex Analysis in General Vector Spaces*. World Scientific, 2002.

[136] M. D. Zeiler, "AdaDelta: An adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[137] L. Zhang, M. Mahdavi, and R. Jin, "Linear convergence with condition number independent access of full gradients," in *Advances in Neural Information Processing Systems*, vol. 26, 2013, pp. 980–988.

[138] T. Zhang and F. J. Oles, "Text categorization based on regularized linear classification methods," *Information Retrieval*, vol. 4, pp. 5–31, 2001.